

Article

Cryptographic DoS Amplification in Hybrid Post-Quantum Deployments: Adversarial Algorithm Substitution and Its Countermeasures

Nazmus Salehin Sammo, Sariya Akhter Lura, Raza Nowrozy and Savitri Bevinakoppa *

School of IT and Engineering, Melbourne Institute of Technology (MIT Melbourne),
Melbourne, VIC 3000, Australia; mit252100@stud.mit.edu.au (N.S.S.); mit252081@stud.mit.edu.au (S.A.L.);
rnowrozy@mit.edu.au (R.N.)

* Correspondence: sbevinakoppa@mit.edu.au

Abstract

Hybrid post-quantum TLS deployments pair classical and post-quantum signature algorithms to resist future quantum adversaries. However, to the best of our knowledge, no active IETF TLS, PQUIP, or LAMPS working-group draft treats CPU-cost adversarial algorithm substitution as a first-class availability threat. We define the Cryptographic Amplification Factor (CAF), a metric for the per-resource cost asymmetry that an adversary induces by forcing a TLS 1.3 server onto a high-cost signature algorithm (SLH-DSA) instead of a low-cost one (ECDSA or ML-DSA). We characterise four adversary classes—insider misconfiguration, supply-chain compromise, remote TLS peer, and protocol man-in-the-middle—demonstrate working proof-of-concept exploits on representative testbeds, and evaluate three defensive primitives under realistic deployment conditions. On the liboqs 0.14.0 reference implementation, $CAF_{r_{cpu}}$ reaches $12,255\times$ on Apple M-series ARM64 and $14,272\times$ on AWS Graviton3 ($3904\text{--}14,272\times$ across five microarchitectures). Under a representative two-worker hardware security module (HSM) calibration ($\lambda = 13.5$ TPS), substitution drives the modelled queue utilisation ρ past the stability boundary ($1.44 \times 10^{-4} \rightarrow 1.787$), a model-independent saturation; on the testbed, we observe server-CPU saturation, accept-queue exhaustion, and dropped connections. Two enforcement defences and a Parallel Signing Architecture provably bound $\rho \leq \alpha < 1$ by admission control, regardless of attack rate. The exposed surface is concentrated today in high-assurance and financial-sector deployments and widens as FIPS 206 finalises.

Keywords: post-quantum cryptography; TLS 1.3; denial-of-service; SLH-DSA; cryptographic amplification factor; hybrid deployment; algorithm substitution; HSM queue saturation



Academic Editor: George A. Tsihrintzis

Received: 16 June 2026

Revised: 9 July 2026

Accepted: 18 July 2026

Published: 23 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Post-quantum cryptography is in production, not on the horizon. As of October 2025, more than half of human-initiated Cloudflare TLS traffic uses hybrid PQ key exchange [1], NIST has finalised FIPS 203–205 [2–4], and national regulators have published migration timelines through 2028–2035. Deployment is outpacing its security analysis (Section 2.2 details the landscape).

The PQ-TLS literature has treated deployment as an *operational* problem—handshake latency, bandwidth, certificate size [5–8]—and IETF drafts address those same concerns [9–13]. The *availability* dimension of algorithm selection has not been treated as a first-class threat.

This paper studies *CPU-cost-asymmetry adversarial algorithm substitution*: an attacker who can influence algorithm selection in a hybrid-PQ negotiation forces the victim onto a high-cost signature algorithm (SLH-DSA) when a low-cost one (ECDSA, ML-DSA) would have sufficed. To the best of our knowledge, as of October 2025, the threat appears in no active IETF TLS/PQUIP/LAMPS working-group draft we surveyed (Section 9), in no regulator advisory, and in no published academic work on hybrid-PQ deployment. The attacker pays little; the victim pays two to three orders of magnitude more than classical handshake-asymmetry attacks [14,15].

Contributions

To the best of our knowledge, no prior work systematically quantifies CPU-cost adversarial signature substitution in hybrid PQ TLS or proposes queue-bounding defences for it. This paper contributes the following:

1. **The Cryptographic Amplification Factor (Section 3).** A per-resource metric (Definition 1) for algorithm-choice asymmetry in hybrid-PQ negotiation, with four realism conditions that bound when $\text{CAF}_{r_{\text{cpu}}} \gg 1$ implies operational denial-of-service.
2. **Threat-class characterisation (Section 4).** Four adversary classes (insider, supply-chain, remote peer, protocol-MITM) with required privileges, plausibility, and observability stated explicitly.
3. **Working PoCs (Section 5).** A1 (insider/misconfig) on an Australian New Payments Platform (NPP) testbed [16] and A3 (remote TLS peer) on Open Quantum Safe (OQS)-OpenSSL 3 with oqs-provider; A2 (supply-chain) modelled.
4. **Measured defences (Section 7).** Two enforcement primitives (D1 selection filtering, D2 algorithm-keyed rate limiting) and the Parallel Signing Architecture, which bound $\rho_{\text{SLH-DSA}} \leq \alpha < 1$ by admission control, plus an early-warning layer (D3 CAF-spike detection).
5. **Honest threat-surface bound (Section 8).** An explicit map of which deployments are exposed today (BSI/ANSSI/NCSC high-assurance, software/firmware signing, financial-sector PQ migration) and which are not (mainline web TLS—all KEM-only). The surface is *pre-deployment*: the correct moment for systematic analysis.
6. **Standards-gap argument (Section 9).** We articulate the gap in active IETF TLS/PQUIP/LAMPS working-group coverage of this threat class and provide the metric and defence vocabulary to fill it (detailed evidence in Section 9).

Illustrative magnitudes

On our testbed, $\text{CAF}_{r_{\text{cpu}}}(\text{SLH-DSA-SHA2-128s, ECDSA-P256}) = 12,255 \times$ (Table 1), driving queue utilisation ρ from $\sim 1.44 \times 10^{-4}$ to $1.787 > 1$ on a two-worker HSM at NPP load ($\lambda = 13.5$ TPS). Larger amplifications have been reported (KeyTrap: $\sim 2 \times 10^6 \times$ [17]); our contribution is not the magnitude but the *attack class*—inter-algorithm substitution in hybrid-PQ negotiation—demonstrated as operationally exploitable with deployable defences. The attack is an *emerging deployment risk*, not a broadly deployed threat; Section 8 bounds the exposed surface in full.

Responsible disclosure

The attack exploits intended TLS 1.3 behaviour, so no protocol CVE applies. We notified the affected maintainers (the OQS project and Entrust) and the IETF TLS working group of the implementation defaults and mitigations discussed here; details are in Section 6.

Reproducibility

All measurement pipelines, attack PoCs (against author-controlled testbed infrastructure only), defence implementations, and data-reduction scripts are available from the corresponding author on reasonable request, with the attack proof-of-concept code shared with verified researchers in line with responsible-disclosure practice.

Table 1. Headline $\text{CAF}_{\text{r}_{\text{cpu}}}$ values on the Apple M-series ARM64 testbed (liboqs 0.14.0/liboqs-python 0.14.1). Baseline: ECDSA-P256, median 21.4 μs ($n = 50$, 5 warmup discarded). All PQC algorithms: $n = 50$, 5 warmup iterations discarded, 1 KiB payload, `time.perf_counter_ns`. Displayed timing values rounded to 1 d.p.; $\text{CAF}_{\text{r}_{\text{cpu}}}$ computed from unrounded medians (ECDSA-P256 precise median 21.375 μs ; $261,945/21.375 = 12,255\times$).

Algorithm	NIST Std.	Median (μs)	p99 (μs)	$\text{CAF}_{\text{r}_{\text{cpu}}}$	Notes
ECDSA-P256	— (baseline)	21.4	28.8	1 $\times$	pyca/OpenSSL 3
ML-DSA-44	FIPS 204	38.7	123.8	1.8 $\times$	lattice; low CAF
ML-DSA-65	FIPS 204	60.4	312.3	2.8 $\times$	lattice; low CAF
ML-DSA-87	FIPS 204	64.1	238.6	3.0 $\times$	lattice; low CAF
FN-DSA-512	FIPS 206 draft	113.2	126.2	5.3 $\times$	Falcon-512
FN-DSA-1024	FIPS 206 draft	227.7	312.5	10.7 $\times$	Falcon-1024
SLH-DSA-SHA2-128s	FIPS 205	261,945	265,076	12,255 $\times$	Headline [†]

[†]Apple M-series figures are illustrative ($n = 50$). Cloud measurements at $n = 500$ in Table A1.

Roadmap

Section 2 covers background; Section 3 defines CAF; Sections 4–7 model, exploit, and defend; Section 8 bounds the threat surface; Section 9 positions the work; Section 10 states limitations; Section 11 concludes.

2. Background

This section summarises the four threads of prior work the paper builds on: the NIST post-quantum standards and their performance profile (Section 2.1); hybrid PQ deployment in TLS 1.3 and the Open Quantum Safe stack (Section 2.2); classical SSL/TLS computational-asymmetry attacks (Section 2.4); and the M/M/c queueing model we use to link per-operation cost to service-level saturation (Section 2.5).

2.1. Post-Quantum Cryptography Standardisation

NIST finalised three post-quantum standards in August 2024: FIPS 203 [2] (ML-KEM, a Module-Lattice-Based Key-Encapsulation Mechanism derived from Kyber), FIPS 204 [3] (ML-DSA, a Module-Lattice-Based Digital Signature Algorithm derived from Dilithium), and FIPS 205 [4] (SLH-DSA, a Stateless Hash-Based Digital Signature Standard derived from SPHINCS+). A fourth standard, FIPS 206 for FN-DSA (derived from Falcon), is expected to be finalised in late 2026 or early 2027.

These algorithms span four orders of magnitude in signing cost. On commodity x86 at 3 GHz, SLH-DSA-SHA2-128s takes $\sim 2.7 \times 10^8$ cycles (~ 90 ms) and -128f $\sim 4.9 \times 10^6$ (~ 1.6 ms) [18]; ML-DSA-65 takes $\sim 3 \times 10^5$ (~ 100 μs). On our Apple M-series ARM64 testbed, ML-DSA-65 signs in 60.4 μs ($n = 50$), giving $\text{CAF}_{\text{r}_{\text{cpu}}}(\text{ML-DSA-65, ECDSA-P256}) \approx 2.8\times$ (ECDSA baseline 21.4 μs ; Table 1); on cloud x86-64, it is 1.8–2.8 $\times$ (Table A1), varying mainly with ECDSA latency (21 μs ARM64 vs. 35–40 μs x86-64), not ML-DSA vectorisation. The SLH-DSA-to-ECDSA cost ratio stays in the 10^3 – 10^4 range on the same hardware—the raw material of the threat model and the CAF metric (Sections 3 and 4).

SLH-DSA's cost is structural. It signs a message digest with a FORS few-time signature whose public key is authenticated through a multi-layer WOTS⁺ hypertree (a tree of Merkle trees), so signing is dominated by sequential hash computation along the hypertree

authentication paths—work the vector and bignum primitives that speed ECDSA and ML-DSA cannot accelerate. The small-signature variants trade signing speed for signature size through taller hypertrees, which is why SLH-DSA-SHA2-128s is far slower to sign than the fast SLH-DSA-SHA2-128f variant (Table A1). This makes SLH-DSA NIST’s canonical conservative hash-based signature: its existential unforgeability reduces to the second-preimage and target-collision resistance of its hash functions, deliberately avoiding reliance on full collision resistance, which is the source of its conservative security margin. It also makes SLH-DSA the worst-case target of the algorithm-substitution attack we study.

Signature Sizes

SLH-DSA-SHA2-128s signatures are 7856 bytes; -128f are 17,088 bytes. ML-DSA-65 signatures are 3309 bytes; public keys are 1952 bytes [18]. The bandwidth-amplification consequences of these sizes are the subject of [12,13,19], which we return to in Section 9; our paper addresses the orthogonal compute-amplification axis.

2.2. Hybrid PQ Deployment in TLS 1.3

Production hybrid PQ TLS is dominated by key exchange: as of October 2025, over 50% of Cloudflare traffic uses X25519MLKEM768 [1]; Chrome shipped it (after middlebox-interop trouble with the earlier X25519Kyber768) [20], and AWS KMS/ACM/Secrets Manager reached GA in 2024–2025 [21]. Signature-side deployment lags: Cloudflare reports “not a single public post-quantum certificate” [1], though AWS added ML-DSA signing and Private-CA code signing in 2025 [22,23], and Microsoft’s SymCrypt ships ML-KEM/XMSS with ML-DSA/SLH-DSA forthcoming [24]. HSM vendors moved faster: Thales Luna 7.9 ships ML-DSA/ML-KEM [25], Entrust nShield v13.8.0 is the only HSM with Cryptographic Algorithm Validation Program (CAVP)-validated native SLH-DSA [26], and Futorex reached PCI HSM PQC validation [27].

Standards Activity

The IETF’s active drafts for hybrid PQ in TLS 1.3 [9–11] define the normative negotiation surface the adversarial substitution of Section 4 exploits. LAMPS drafts [28–30] specify the X.509 identifiers for the signature algorithms involved. None of these drafts treats adversarial algorithm selection as a first-class availability threat; we develop the gap in Section 9.

2.3. The Open Quantum Safe Stack

Our PoC targets the Open Quantum Safe (OQS) stack [7]: the liboqs C library providing prototype implementations of NIST PQ primitives, the oqs-provider OpenSSL 3 provider exposing those primitives through TLS 1.3, and the OQS-OpenSSL 3 test-server build that integrates them end-to-end. This stack is the substrate for published PQ TLS performance work [5,6]. The OQS project positions it for research and experimentation, not production deployment; the exposure we characterise is therefore a property of the *default configuration* that a production hybrid-PQ implementation adopting comparable settings would inherit, with OQS as our measurement substrate. Its prior disclosure history (e.g., CVE-2024-37305, a heap overflow in hybrid-key deserialisation [31], patched in v0.6.1) is availability-impacting but at the implementation layer, not the algorithm-selection layer we address. We pin liboqs 0.14.0 (Python 3.14.4 shim benchmarks; cloud runs on Python 3.10.12) and oqs-provider 0.11.0/OpenSSL 3.6.0 (the openquantumsafe/oqs-oss13 Docker image, for the TLS measurements)—the latest stable tags at the time of this work.

2.4. Classical SSL/TLS Computational-Asymmetry DoS

Two canonical tools define the classical baseline: sslsqueeze [14] and THC-SSL-DOS [15]. Both exploit the server-side asymmetry of an RSA-2048 (or similar) handshake:

the server performs private-key work the client cannot pre-empt. Reported amplification ratios are in the $6\text{--}15\times$ range on legacy configurations; Bernat [32] surveyed mitigations (client puzzles, RSA $\rightarrow$ ECDHE migration, algorithm-keyed rate limiting). The amplification is bounded by the ratio of server-side private-key operations to the trivial client-side work of issuing a handshake. These attacks establish the threat class CAF generalises—attacker-induced asymmetry at the cryptographic layer—which SLH-DSA extends by two to three further orders of magnitude once it enters the negotiation surface.

2.5. HSM Queueing and the M/M/c Model

Per-operation cost alone does not produce denial-of-service; queue dynamics at bounded concurrency are what drive saturation. Our analysis uses the standard M/M/c model: Poisson arrivals at rate λ , exponential service at rate μ per server, and c parallel servers, giving utilisation $\rho = \lambda/(c\mu)$. Queue stability requires $\rho < 1$; once $\rho \geq 1$, the queue grows without bound at rate $\lambda - c\mu$, which is the only property the saturation argument needs. (Below the boundary, mean waiting time follows the Erlang-C formula.)

Substituting an SLH-DSA service rate for an ECDSA one at a fixed arrival rate shifts ρ by the factor of the per-operation cost ratio. At the Australian NPP daily-average arrival rate of $\lambda \approx 13.5$ TPS on a $c = 2$ HSM configuration, $\rho_{\text{ECDSA-P256}} \approx 1.44 \times 10^{-4}$ and $\rho_{\text{SLH-DSA-SHA2-128s}} = 1.787 > 1$ (measured; $N = 8$ settled trials; see Section 5.1), crossing the stability boundary into unbounded queue growth. The ratio $\rho_{\text{SLH-DSA-SHA2-128s}}/\rho_{\text{ECDSA-P256}} \approx 12,400\times$ is the queue-utilisation variant of CAF for that scenario, consistent with the headline CPU-level figure (Table 1); the small difference reflects shim-layer HTTP overhead in the service-rate measurements.

Limitation of the Analytical Model

This ratio derives from software-crypto latencies on Apple M-series ARM64, not dedicated HSM silicon; Section 5 reports end-to-end SoftHSMv2 values, and Section 10 discusses HSM realism and the service-time assumption (SLH-DSA signing is near-deterministic, so M/D/c strictly applies, but, for $\rho > 1$, both models predict unbounded growth—the saturation claim is model-independent).

2.6. Service-Time and Arrival-Process Fidelity

M/M/c assumes exponential service and Poisson arrivals. SLH-DSA signing violates the first—it is near-deterministic (coefficient of variation $C_s = \text{SD}/\text{mean} \approx 5098/261,945 = 0.019$, so $C_s^2 \approx 3.8 \times 10^{-4}$; Section 5.1)—and real payment traffic violates the second, being burstier than Poisson. We quantify both deviations; neither affects the saturation result.

Saturation is distribution-free. The attack hinges on $\rho > 1$, i.e., $\lambda > c\mu$. Once the offered rate exceeds the aggregate service rate, the backlog of any work-conserving queue grows without bound at rate $\lambda - c\mu$, independent of the arrival and service distributions: cumulative departures from a work-conserving c -server queue cannot exceed $c\mu t$ over any interval of length t , so, once the long-run offered rate λ —however bursty or non-stationary the adversarial arrival stream—exceeds $c\mu$, the backlog grows at least linearly at rate $\lambda - c\mu$ by simple rate balance, requiring no ergodicity assumption on the input. Loynes' criterion [33] supplies the matching *stability* result for the $\rho < 1$ regime under stationary-ergodic input. The headline $\rho = 1.787 > 1$ therefore holds identically under M/M/c, M/D/c, and any G/G/c refinement—only the *sub-saturation* mean queue length is distribution-sensitive.

Deterministic service (the M/D/c correction). Below saturation, the Pollaczek–Khinchine formula scales the mean waiting time by $(1 + C_s^2)/2$ relative to the memoryless case [34]. With $C_s^2 \approx 3.8 \times 10^{-4}$, this factor is ≈ 0.5 : M/D/c mean waiting time is about half the M/M/c value at the same ρ (exact for a single server; the multi-server extension is the standard Allen–Cunneen approximation, whose error is immaterial to our argument:

with near-deterministic service the exact $M/D/c$ mean wait likewise stays below the $M/M/c$ value for every $\rho < 1$, so the *sign* of the correction— $M/M/c$ over-predicts—does not depend on the approximation). This single-server exactness is not merely hypothetical: the PSA SLH-DSA isolation lane is provisioned with $c' = 1$ (Section 7.4, Proposition 1), the regime in which Pollaczek–Khinchine is exact, so there the true mean wait is $(1 + C_s^2)/2 \approx 0.5002$ of the $M/M/1$ value and $M/M/c$ over-predicts queueing by $\approx 1.9992 \times$ —a residual model error below 0.04%. Every admission threshold we derive from $M/M/c$ is thus conservative by this known, bounded factor rather than an unquantified one. $M/M/c$ is therefore *conservative* in the defended regime—it over-predicts queueing by $\approx 2 \times$, so the admission and rate-limit thresholds we derive from it (Section 7) bound ρ earlier than strictly necessary, which is the safe direction for a defender.

Bursty arrivals (the opposing correction). Burstiness pushes the other way. The Allen–Cunneen $G/G/c$ approximation scales the $M/M/c$ wait by $(C_a^2 + C_s^2)/2$, where C_a^2 is the squared coefficient of variation of inter-arrival times [34]. For near-deterministic service, this is $\approx C_a^2/2$: Poisson arrivals ($C_a^2 = 1$) recover the 0.5 $M/D/c$ factor, whereas strongly bursty load ($C_a^2 > 2$) can exceed the $M/M/c$ prediction—the net correction for our setting is $(C_a^2 + 3.8 \times 10^{-4})/2$. We retain $M/M/c$ for its closed-form Erlang-C bounds: it is accurate to a small constant factor for moderately bursty traffic and conservative under Poisson or smoother load, while heavier burstiness only lengthens the true queue and strengthens, rather than weakens, the availability concern.

3. The Cryptographic Amplification Factor

This section defines the Cryptographic Amplification Factor (CAF), states its per-resource and aggregate forms, gives the operational measurement protocol, and places the metric against prior work. The definition is a specialisation of the CPU amplification factor of QFAM [35], a defence against handshake flooding in the QUIC transport protocol, applied to hybrid post-quantum signature substitution, framed around the victim’s cryptographic-operation queue rather than per-connection CPU accounting.

3.1. Notation and Resources

Let V be a victim performing a cryptographic operation op with algorithm A from the deployment’s supported set S_V , and let $\mathcal{A}$ be an adversary class (A1–A4, Section 4) able to influence V ’s choice of A . We instantiate CAF across five resource dimensions r : CPU (r_{cpu} , $\mu\text{s/op}$ on a pinned core), memory (r_{mem} , peak resident bytes), bandwidth (r_{bw} , wire bytes/op), queue (r_{queue} , $1/\mu$ on the cryptographic-operation queue), and energy (r_{energy} , $\mu\text{J/op}$).

3.2. Per-Resource CAF

Definition 1 (Per-resource CAF). Let $vc_r(op, A; V)$ denote the resource- r cost induced on V when it performs op with algorithm A , and $ac_r(\text{trig}, A; V)$ the resource- r cost to the adversary of one trigger causing V to perform that operation. For a baseline algorithm $B \in S_V$,

$$\text{CAF}_r(A, B; V) = \frac{vc_r(op, A; V)/ac_r(\text{trig}, A; V)}{vc_r(op, B; V)/ac_r(\text{trig}, B; V)}. \quad (1)$$

Intuitively, $\text{CAF}_r > 1$ means the adversary’s choice of A over B shifts more resource- r cost onto the victim per unit of attacker effort than the baseline does. The units of vc_r and ac_r must match, so CAF is dimensionless.

Practical simplification. When the adversary’s per-trigger cost is identical for both algorithms (as in A1, with the same HTTP PUT differing only in the algorithm name, and A3, with the same ClientHello structure but different extension content), $ac_r(\text{trig}, A; V) = ac_r(\text{trig}, B; V)$ and CAF reduces to the victim cost ratio $vc_r(op, A; V)/vc_r(op, B; V)$. The ratio-of-ratios

form is retained because ac_r can differ substantially across adversary classes (A2's library-distribution infrastructure cost amortises to near-zero per trigger, as discussed in Section 5.3), and the general form makes these differences explicit rather than absorbing them into assumptions.

3.3. Reporting: The CPU Dimension

A deployment exposes several candidate algorithms across several resource dimensions (CPU, memory, bandwidth, queue-seconds, energy), and an adversary picks the algorithm and dimension that maximise the amplification it can realise. Throughout this paper, we report the CPU-dimension figure $CAF_{r_{\text{cpu}}}$, because CPU-queue saturation is the operative threat: the HSM's CPU pool is the scarce resource, and queue stability requires $\rho = \lambda / (c\mu) < 1$. The other dimensions have far smaller per-operation ratios for SLH-DSA (certificate-size growth is only $\sim 16\times$ over ECDSA, not $10^4\times$), so the single CPU figure is both the relevant and the conservative choice for the saturation argument.

Concretely, the per-operation amplifications span very different magnitudes. CPU is $10^3\text{--}10^4\times$ (the headline $12,255\times$). Wire bandwidth is $\approx 10^2\times$, set by the CertificateVerify signature the server emits per handshake (7856 B for SLH-DSA-SHA2-128s versus ≈ 72 B for an ECDSA-P256 signature); this is the second-largest axis but still two orders of magnitude below CPU. Peak resident memory and steady-state energy per operation are modest, the latter tracking CPU time to first order. CPU therefore both dominates and is the resource gated by the bounded HSM queue, which is why it is the operative and conservative axis; the remaining dimensions are reported per-resource by Definition 1 and discussed as a coupled model in Section 3.10.

3.4. Adversary Cost Conventions

We include in ac_r only the costs that the adversary incurs *per trigger*: bandwidth to deliver the trigger, compute to construct it, and any per-trigger crypto cost (e.g., the attacker's own KEM share where reciprocity is required). We exclude amortisable setup costs (key generation, certificate enrolment), matching the convention in the classical SSL-DoS literature [14,15,32].

3.5. Threat Realism Conditions

$CAF_{r_{\text{cpu}}} \gg 1$ is necessary but not sufficient for an operational denial-of-service based on CPU-queue saturation. The following four conditions must additionally hold; we report each alongside $CAF_{r_{\text{cpu}}}$ for every measured (deployment, algorithm) pair so readers can independently judge whether the threat is operational or purely arithmetic:

1. **Algorithm in supported set.** $A \in \mathcal{S}_V$. The deployment must actually negotiate the high-CAF algorithm.
2. **Adversary can influence selection.** The adversary class $\mathcal{A}$ reaches the selection surface (A1/A2/A3/A4 per Section 4).
3. **Victim has utilisation headroom.** $\rho_V(B_V^*) < 1$. A victim already at saturation under the baseline algorithm cannot be amplified further by definition.
4. **Per-operation cost dominates per-trigger cost.** $vc_r(\text{op}, A; V) \gg ac_r(\text{trig}, A; V)$ when $r = r_{\text{cpu}}$. This is not circular: for SLH-DSA-SHA2-128s in the A3 setting, the victim pays 262 ms of CPU time while the attacker pays $\lesssim 0.1$ ms to construct and send a ClientHello (≈ 300 bytes). The per-operation ratio is $\approx 2600\times$, confirming the attack is compute-bound. For a pure-bandwidth amplification attack (e.g., large-certificate TLS), this condition would fail and bandwidth-side defences (certificate compression) would be the correct framing instead.

3.6. Why a New Metric (and Its Relation to Prior Work)

Why existing amplification metrics do not fit

Classical SSL-DoS amplification [14,15] is a single-algorithm, single-resource ratio: server work over client work for a *fixed* cipher suite. It can express “an RSA-2048 handshake costs the server $10\times$ the client,” but it has no free variable for the algorithm the victim is made to run, so it cannot express “the adversary forced RSA-2048 *instead of* ECDHE.” The hybrid-PQ asymmetry is exactly a ratio of two such ratios across an adversary-controlled algorithm choice: the free variable A against baseline B in Definition 1. The ratio-of-ratios is therefore not ceremony but the minimal form that makes the adversary’s *selection*, rather than a fixed handshake’s cost, the quantity measured. The new analytic step over both classical SSL-DoS and QFAM is the link from this per-operation ratio to a queue-stability boundary through $\rho = \lambda/(c\mu)$ (Section 2.5), which turns a microbenchmark into an availability predicate. We state the novelty precisely, conceding what is *not* new: that SLH-DSA is $\sim 10^4\times$ slower than ECDSA is a measured benchmark [4,18], and that $\rho = \lambda/(c\mu)$ governs queue stability is textbook. The contribution is neither ingredient but their coupling under an *adversary-controlled* free variable: in hybrid-PQ negotiation, the signing scheme is selected per handshake by an unauthenticated peer (Section 4.4), so a static per-suite cost benchmark becomes a *reachable* availability predicate rather than a fixed operational figure. The four realism conditions (Section 3.5) are what separate this reachability claim from arithmetic, and are absent from every prior amplification metric we surveyed—including QFAM.

Relation to QFAM [35]

QFAM defines a CPU amplification factor for QUIC handshake flooding on classical cipher suites, reporting $\sim 4.6\times$ on aioquic. CAF adopts the name and the per-operation idea—we cite QFAM as the conceptual origin—but differs on three DoS-relevant axes (Table 2): a far wider cost spread (10^3 – $10^4\times$ for NIST signatures vs. sub- $10\times$ for cipher suites); a lift from a per-connection ratio to the service-rate utilisation ρ that actually produces saturation; and the four realism conditions (Section 3.5) that separate “arithmetic” from “operational” amplification, which QFAM does not formalise.

Table 2. Why CAF is needed: classical SSL-DoS and QFAM cannot express adversary-selected, queue-coupled signature asymmetry.

Property	Classical SSL-DoS	QFAM	CAF (This Work)
Algorithm a free variable?	No (fixed suite)	Partial (cipher suite)	Yes ($A \in \mathcal{S}_V$, signatures)
Cost spread exploited	6–15 $\times$	$\sim 4.6\times$	10^3 – $10^4\times$ (SLH-DSA/ECDSA)
Resource dimensions	CPU only	CPU only	CPU, mem, bw, queue, energy
Links to queue saturation?	No	No	Yes ($r_{\text{queue}} + M/M/c\rho$)
Arithmetic-vs-operational gate	Implicit	None	Four realism conditions

Relation to KeyTrap and client puzzles

Two further quantities are adjacent. KeyTrap [17] measures intra-algorithm validation-work amplification on DNSSEC resolvers; client-puzzle difficulty [36,37] formalises *defender-imposed* asymmetry, the mirror image of the *attacker-induced* asymmetry CAF measures. We treat both in full in Section 9, where the distinguishing contrasts (inter- versus intra-algorithm, and attacker- versus defender-imposed) are developed.

3.7. Reference Algorithm Set

We report CAF for ML-DSA-{44,65,87} [3], representative SLH-DSA variants [4], and FN-DSA-{512,1024} (FIPS 206 draft) against the ECDSA-P256 baseline; ML-KEM [2] and RSA-2048 data are in the artefact. The canonical table across all five microarchitectures ($n = 50/500$, liboqs 0.14.0) is Table A1; headline numbers are in Table 1.

Headline

On the NPP-representative testbed, $\text{CAF}_{r_{\text{cpu}}}(\text{SLH-DSA-SHA2-128s, ECDSA-P256}) = 12,255 \times$ (liboqs 0.14.0, Apple M-series ARM64 (Apple M3 Max); $n = 50$, 5 warmup iterations discarded; SLH-DSA median 261,945 μs , ECDSA median 21.375 μs (displayed 21.4 μs); `expanded_bench.py`, 18 April 2026). The queue-stability variant confirms $\rho = 1.787 > 1$ at the working calibration $\lambda = 13.5$ TPS with $c = 2$ HSM workers (Section 5.1). Independent re-runs across $N = 8$ settled trials produced $\text{CAF}_{r_{\text{cpu}}}$ values within $\approx 2\%$ of the headline figure, indicating low run-to-run variance for SLH-DSA on the ARM64 scalar path. We report the ARM64 value as the headline for continuity with our primary (Apple M-series) testbed, but it sits near the top of the measured $3904\text{--}14,272 \times$ range (Table A1, second only to Graviton3's $14,272 \times$): on the three x86-64 server microarchitectures that dominate production TLS termination, $\text{CAF}_{r_{\text{cpu}}}$ is $3904\text{--}4389 \times$. The ARM64 magnitude is amplified partly by an unusually fast ECDSA denominator (21.4 μs versus 35–40 μs on x86-64), not by slower SLH-DSA. The saturation argument holds identically at the more conservative $\sim 4000 \times$: SLH-DSA service capacity $c\mu = 7.55 \ll \lambda = 13.5$ regardless of which baseline sets the ratio. ML-DSA $\text{CAF}_{r_{\text{cpu}}}$ is substantially lower ($1.4\text{--}6.2 \times$ across all variants and architectures); Section 8.6 explains why excluding SLH-DSA from deployments is not a sufficient defence despite this difference.

3.8. Comparison to Classical DoS Asymmetries

We place CAF for the SLH-DSA-vs-ECDSA case against four reference DoS classes: classical SSL computational asymmetry [14,15] ($6\text{--}15 \times$), HashMap collision DoS [38] ($\sim 10^2\text{--}10^4 \times$), Memcached UDP reflection ($\sim 50,000 \times$), and KeyTrap DNSSEC validation explosion [17] ($\sim 2,000,000 \times$).

The *mechanism* is what distinguishes this work: hybrid-PQ algorithm substitution introduces a per-operation cost asymmetry two to three orders of magnitude above classical SSL asymmetry, at a point in the TLS handshake where prior work has not treated adversarial selection as a first-class threat (Section 9). Cost asymmetry between operations is studied outside security as well, for example, in steganographic colour conversion [39], where an embedding scheme trades encoding cost for imperceptibility. That setting differs from ours in kind: the overhead there is an offline, single-party design choice, whereas the asymmetry we measure is induced by an adversary at a protocol negotiation point and gated by a queue-stability boundary.

3.9. Measurement Protocol

Two measurement types are used and must not be conflated. The *isolated per-operation CPU cost* times only the cryptographic primitive (the `sign()` call, key generation hoisted out of the loop); this is the input to $\text{CAF}_{r_{\text{cpu}}}$ and the queue model. The *end-to-end TLS wall-clock time* measures a full handshake at the client and is reported only to show that signing dominates the handshake, never as the $\text{CAF}_{r_{\text{cpu}}}$ input (Section 5.2). Warm-up invocations are discarded; signing is timed via `time.perf_counter_ns` ($n = 50$ on the ARM64 baseline, $n = 500$ per cloud microarchitecture under a performance governor with Intel turbo disabled); we report median (the headline CAF) and p99. The full step-by-

step protocol—governor settings, getrusage accounting, and the per-class adversary-cost harnesses—and all pipelines are available on request.

3.10. What CAF Does Not Capture

We explicitly enumerate what the metric omits:

- **Cache-line contention between concurrent operations.** Our measurements are single-core; multi-core contention under high-throughput SLH-DSA signing is a separate phenomenon and would amplify the observed CAF, not reduce it. We leave quantitative modelling to future work.
- **Downstream cascading failures.** A saturated HSM queue can trigger retries at higher layers (application, client, interbank settlement), amplifying the visible impact well beyond the queue. Our evaluation reports only the queue-level impact; the systemic view is scope for a follow-up.
- **Energy cost under thermal throttling.** We report microjoules on steady-state thermal envelope; sustained SLH-DSA floods can induce throttling that further inflates per-operation cost. Future work could expand, since we report only the steady-state figure.
- **Key-management overhead.** We do not model the additional KMS/HSM-key-lifecycle cost of maintaining PQ key material. This is relevant for cost-modelling but orthogonal to the per-operation DoS metric.
- **Coupled multi-resource amplification.** We report each resource dimension's CAF independently (Definition 1). Under a simultaneous flood, the dimensions interact—one SLH-DSA signature consumes wire bandwidth, output-buffer memory, and the hashing CPU that produces it at once—so the realised multi-resource pressure can exceed any single-axis figure. A full coupled model over the resource vector, with cross-terms, is left to future work; a conservative bottleneck bound, however, follows without one. The dimensions are incommensurable cost ratios (Definition 1), so a simultaneous flood does not sum them—what binds the victim is the scarcest resource. For the bounded HSM CPU pool, the operative constraint is $\rho = \lambda / (c\mu) < 1$ (Section 2.5) on the worst axis, which no other axis can relieve, so coupled amplification is governed by the dominant axis, $\text{CAF}_{\text{coupled}} = \max_r \text{CAF}_r$ —the CPU value of $12,255\times$, since wire bandwidth ($\approx 10^2\times$), memory, and energy sit two or more orders of magnitude below and cannot bind before CPU saturates. The CPU axis we report is thus its dominant and conservative coupled component.

4. Threat Model

4.1. Deployment Scope (Summary)

The attack surface described here is *latent today and widening*: it applies immediately to BSI/ANSSI/NCSC-aligned high-assurance deployments, financial-sector PQ migration programmes, and software/firmware signing infrastructure that exposes a TLS signing endpoint—wherever an SLH-DSA (FIPS 205) leaf certificate can appear in a negotiated cert chain. It does *not* apply to mainline web TLS (Cloudflare, Chrome, most content delivery network (CDN)-terminated traffic), which as of April 2026 carries no PQ signatures. NSA CNSA 2.0 restricts SLH-DSA to firmware signing, narrowing the US-government NSS surface. Section 8 gives the full in-scope and out-of-scope taxonomy.

The threat does not depend on a strong adversary: four independent classes reach the algorithm-selection surface, and the central claim of this section is that even the weakest of them—A3, an ordinary remote TLS client with no credentials—suffices to drive the CAF we measure. We describe four adversary classes that can induce a hybrid-PQ negotiation to select a high-CAF signature algorithm. For each, we state required privilege, plausibility, observability by standard defensive tooling, and the subset of the paper's PoC work

that targets it. We present these attributes under a fixed labelled rubric—rather than as free prose—by deliberate design, so the adversary classes can be compared attribute-by-attribute across A1–A4; the labels are a structured comparison schema, not a substitute for analysis, and each is expanded in a full sentence below. The classes are ordered from high-privilege/low-reach (A1 misconfiguration/insider) to low-privilege/high-reach (A3 remote TLS peer, A4 MITM), to highlight that, in hybrid-PQ deployments, even the lowest-privilege adversary (A3) can drive the CAF we measure. Figure 1 visualises the A3 attack flow and its queueing consequence.

The practicality of this threat class is corroborated by independent PQ protocol work. Darzi et al. [40] build explicit DoS-resilience mechanisms into a post-quantum private spectrum-access protocol, treating adversarially induced computation cost as a first-class availability threat. PQ-protocol designers, in other words, already recognise algorithm-cost asymmetry as an exploitable surface. The defensive primitives evaluated in Section 7 (algorithm-selection filtering and algorithm-keyed rate limiting) are direct extensions of the classical mitigations surveyed by Bernat [32]; their deployability is demonstrated against the testbed configurations in Section 5.

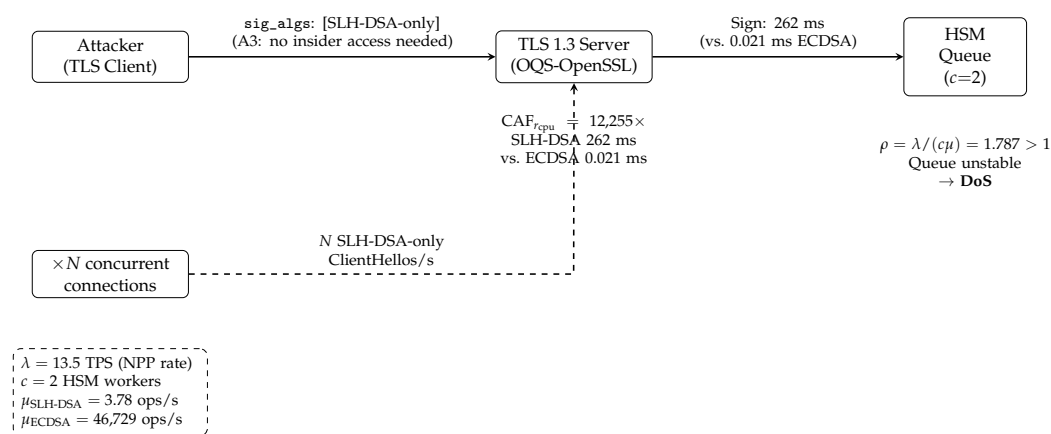


Figure 1. Adversarial algorithm substitution in TLS 1.3 (adversary class A3). A network-level attacker sends a ClientHello advertising only SLH-DSA-SHA2-128s in `signature_algorithms`. The server must sign its CertificateVerify with SLH-DSA or abort; SLH-DSA signing requires 262 ms vs. 0.021 ms for ECDSA ($\text{CAF}_{\text{cpu}} = 12,255\times$). At Australia’s NPP calibration ($\lambda = 13.5$ TPS, $c = 2$ HSM workers), N concurrent SLH-DSA-only connections drive queue utilisation to $\rho = 1.787 > 1$, producing unbounded queue growth and service denial. The attack requires no credentials, no insider access, and no packet modification—only a crafted ClientHello extension.

4.2. A1: Insider/Misconfiguration

Capability

A1 has configuration write access to the HSM’s algorithm-policy surface or the service’s selection logic—a privileged operator, a deploy-script bug, or an unreviewed artefact (e.g., a Helm chart defaulting `signing_algorithm` to `slh_dsa_128s`)—with no special network reach.

Plausibility

High: configuration drift is a recognised operational-risk category in regulated finance (APRA CPS 234 [41] mandates controls precisely because security-setting misconfiguration is a recurring incident class), and SLH-DSA adoption in BSI/ANSSI/NCSC jurisdictions makes its presence in production configurations likely.

Observability

Low: a flipped signing algorithm is invisible to application-layer monitoring unless algorithm-type is instrumented—rarely a first-class metric—so queue-depth alerts fire only after saturation is under way.

PoC

A1 is demonstrated end-to-end in Section 5.1 against the `npp_extension` testbed node running SoftHSMv2 with a deliberately permissive PKCS#11 algorithm-selection interface. The attack primitive is a single HTTP PUT that rewrites the shim’s algorithm parameter from ECDSA-P256 to SLH-DSA-SHA2-128s; queue saturation follows immediately upon algorithm flip, since $\lambda = 13.5 > c\mu_{\text{SLH-DSA-SHA2-128s}} = 7.55$ ops/s at the realistic NPP arrival rate.

4.3. A2: Supply-Chain (Modelled)

A2 modifies an upstream cryptographic library (liboqs, oqs-provider, a distro package, a container image) to alter the default algorithm preference, triggering on any victim that ingests the compromised version. Plausibility is medium: the class has a 2024–2026 track record (xz-utils, npm incidents), and the OQS project has already issued advisories for memory bugs (CVE-2024-37305). Detection relies on reproducible-build verification, software bill of materials (SBOM) analysis, or source review, which smaller organisations rarely run in full. We *model* A2 rather than build a live exploit (which would cross ethical lines): Section 5.3 analyses the path analytically and bounds its induced CAF on a controlled backdoored-preference build.

4.4. A3: Remote TLS Peer

Capability

A3 has no insider access, no credentials, no MITM position. It is a normal TLS 1.3 client with network reach to the victim’s TLS endpoint, able to negotiate a handshake and choose its `signature_algorithms` preference list. In TLS 1.3, the server signs the handshake transcript in `CertificateVerify` with its leaf key, using a scheme drawn from the intersection of the client’s `signature_algorithms` extension (RFC 8446 §4.2.3, §4.4.3) and the schemes its key supports; the leaf certificate’s own issuer signature is separately constrained by `signature_algorithms_cert`, which defaults to `signature_algorithms` when the client omits it. When that intersection contains only SLH-DSA, the server must sign with SLH-DSA or abort, and the costed operation is the server-side `CertificateVerify` signing (Section 5.2).

Plausibility

High on any deployment whose cert chain includes an SLH-DSA leaf certificate (or FN-DSA-1024 (FIPS 206 draft; forthcoming), which has $\text{CAF}_{\text{cpu}} \approx 10.7\times$ over ECDSA on ARM64). This is the *intended* TLS 1.3 behaviour: the negotiation surface gives the client a say. Nothing in the protocol prevents a client from declaring a preference for the most expensive option the server can offer. The only thing preventing CAF-driven exploitation today is that mainline web deployments do not yet include PQ signatures in their cert chains (per Cloudflare’s October 2025 report: “not a single public post-quantum certificate is used” [1]). In BSI/ANSSI/NCSC-aligned high-assurance contexts and in the federated authentication scenarios outlined in the SLH-DSA X.509 LAMPS draft [29], the cert chain will include SLH-DSA.

Observability.

Low: the handshake is indistinguishable at wire level from any other well-formed TLS 1.3 handshake. An algorithm-type-aware logger (collecting the selected server-cert signature algorithm per handshake) can detect the shift, but deployments that log this are rare.

PoC.

A3 is demonstrated end-to-end in Section 5.2 against the `vulnerable_server` testbed node running OQS-OpenSSL 3 with `oqs-provider` configured to advertise SLH-DSA-SHA2-128s in its signature-algorithms list. The attack client is a standard Go `crypto/tls` client (with `oqs-go` bindings for PQ algorithms), advertising only SLH-DSA as its preferred signature algorithm.

4.5. A4: Protocol-Level MITM (Discussed)

A4 sits on-path and strips or reorders TLS extensions, modifying the client's `signature_algorithms` (or exploiting a middlebox handshake-retry fallback) to force a high-CAF selection. Plausibility is medium and bounded: TLS 1.3 authenticates its transcript, which defeats direct MITM tampering (a tampered list causes a visible authentication failure), so A4 reopens only through fallback-retry code paths—of the kind Chrome's v124 rollout exposed [20]—that silently reorder preferences. We *discuss* A4 rather than demonstrate it: its exploitability is deployment-specific and not amenable to a single reproducible PoC. The D1/D2 mitigations that cap A1–A3 also bound A4's amplification once server CPU cost is capped, but its trigger mechanism warrants separate study.

4.6. Adversary Goal

Across all four classes, the adversary's goal is uniform: induce service queue saturation at the cryptographic-operation layer such that $\rho_V \geq 1$ and SLA breach or service denial follows. Equivalently, the goal is $\text{CAF}_{r_{\text{cpu}}}(A, B_V^*; V) \gg 1$ for some $A \in \mathcal{S}_V$ (Definition 1), driving $\rho_V = \lambda / (c\mu_A) \geq 1$ at the cryptographic-operation queue, under the four realism conditions in Section 3.5.

We are explicit that the adversary is not attempting confidentiality or integrity compromise; those threats are covered by the extensive prior literature on TLS downgrade and key-exchange security [42,43]. CAF is an *availability*-side threat metric.

4.7. Threat Realism Conditions

For each concrete (deployment, adversary, algorithm) triple, we evaluate the four realism conditions in Section 3.5 and report the result alongside the numerical $\text{CAF}_{r_{\text{cpu}}}$ in Section 5. A $\text{CAF}_{r_{\text{cpu}}}$ that fails any realism condition is reported as arithmetic; one that passes all four is flagged as operationally exploitable.

4.8. Non-Goals and Out-of-Scope

We explicitly do not consider the following:

- Quantum-capable adversaries. This paper is about classical denial-of-service in the transition-to-PQ period; Shor/Grover adversaries are orthogonal.
- KEM-side CAF as a primary attack surface. KEM-side asymmetry across NIST-standardised algorithms is single-digit and does not produce the order-of-magnitude amplification the signature side does. We include KEM rows in the canonical CAF table for completeness.

- Denial-of-service against identity/authentication beyond TLS (e.g., OpenID-Connect PQ token verification). The threat class generalises but we focus on the TLS handshake where deployment data is most mature.
- Algorithm-negotiation integrity attacks against composite signatures [28]. Composite signatures present a related but distinct threat surface; we sketch the extension in future work.

5. Exploit Demonstrations

The demonstrations that follow share a single quantitative core: the same per-operation cost asymmetry (Table 1) drives each adversary class across the queue-stability boundary $\rho > 1$ at realistic arrival rates, so A1–A3 are three delivery paths to one saturation result rather than three independent findings. This section reports PoCs for the two adversary classes we demonstrate end-to-end, and the modelled analysis of the third. Each subsection follows the same structure: testbed topology, attack chain, measurement protocol, results, and realism-condition summary per Section 3.5.

All PoCs run against author-controlled testbed nodes. No third-party system is exercised.

Loopback execution

The measurements reported below are collected with the vulnerable server and the attack client on the same host, communicating over the loopback interface (native `local` path or Docker bridge). CAF is CPU-side by construction; removing the network-RTT component does not bias the metric, and the loopback configuration is documented in the reproducibility package (available on request). Measurements across geographically distributed origins add a cascading-failure story that is orthogonal to the CAF claim and is treated as future work (Section 10).

5.1. A1: Insider/Misconfiguration on the NPP-Representative Testbed

Setup

The `npp_extension` testbed node runs `SoftHSMv2` initialised with `softhsm_parallel_slots = 2`, matching a minimal two-worker high-availability HSM configuration. Above `SoftHSMv2` sits a thin Python shim exposing an HTTP endpoint whose request body selects the signing algorithm. The shim is a single-threaded synchronous dispatcher: each request is served to completion under the CPython global interpreter lock (GIL), so it adds serialisation and interpreter overhead but no parallelism of its own. That overhead is small and bounded: the shim-mediated SLH-DSA mean of 264,804 μ s exceeds the isolated primitive median of 261,945 μ s by ~ 2859 μ s ($\approx 1.1\%$), two orders of magnitude below the 262 ms signing cost, so the reported queue behaviour is driven by the cryptography, not the harness; the isolated figure—not the shim-mediated one—feeds $\text{CAF}_{r_{\text{cpu}}}$ and the queue model. The shim's `/metrics` endpoint publishes per-algorithm cumulative operation counts and cumulative microseconds, sufficient to reconstruct per-second algorithm-mix and queue utilisation.

The benign generator drives ECDSA-P256 signing at $\lambda \approx 13.5$ TPS, representative of Australian NPP daily-average interbank volumes [16]. The exact value is not load-bearing: the saturation result ($\rho > 1$) holds for any (λ, c) with $\lambda > c\mu_{\text{SLH-DSA}} = 7.55$ ops/s ($\mu = 3.777$, $c = 2$), which any moderate-volume signing endpoint exceeds; we report 13.5 TPS, $c = 2$ as a concrete worked example.

Attack chain

The attack chain is as follows:

1. At $t = 0$, the traffic generator is running steady-state with $\lambda = 13.5$ TPS, $\rho \approx 1.44 \times 10^{-4}$.

2. At $t = T$, the A1 adversary (modelled as a configuration PUT reaching the shim) flips the algorithm parameter from ECDSA-P256 to SLH-DSA-SHA2-128s. No new requests are injected; the existing benign traffic continues at the same arrival rate.
3. The shim now signs each request with SLH-DSA-SHA2-128s (~ 265 ms shim-mediated, 262 ms isolated; Table 1). Service capacity $c\mu = 7.55$ ops/s falls below $\lambda = 13.5$, so ρ crosses the stability boundary and the queue grows without bound. Measured values are reported below.

Measurement

The testbed orchestrator polls the shim's `/metrics` at 100 ms resolution for 60 min (5 baseline, 10 attack, 45 drain). Metrics derived per second: arrival rate, per-algorithm service rate, mean and p99 per-request wall time, and queue depth estimated via Little's law. Ten independent runs; results are reported as mean $\pm$ SD with 95% confidence intervals (Student- t , $df = 7$).

Results

The finding, stated first: flipping only the signing-algorithm parameter—with the benign arrival rate held fixed and no new requests injected—drives ρ from $\sim 1.44 \times 10^{-4}$ to 1.787, past the stability boundary, so the saturation is caused by the substitution itself, not by any added load. Measurements are on Apple M-series (ARM64); liboqs 0.14.0 via liboqs-python 0.14.1. Ten independent runs ($N = 10$): the first two trials exhibit OQS library cold-start overhead (SLH-DSA p50 ≈ 375 ms vs. ≈ 265 ms for settled trials) and are excluded from the CI computation; all ten trial records are retained in the reproducibility package (available on request). Values below are from the eight settled trials ($N = 8$).

Per-operation timing (shim-mediated). SLH-DSA-SHA2-128s shim service time: mean 264,804 μ s (95% CI [260,541, 269,066] μ s; SD 5098 μ s; $N = 8$ settled trials). This slightly exceeds the isolated CPU benchmark (261,945 μ s; Table 1) due to HTTP-layer dispatch and GIL scheduling overhead in the shim. $\text{CAF}_{r_{\text{cpu}}}$ is defined and measured from isolated signing primitives (Definition 1); the canonical value is $12,255 \times$ (Table 1, $n = 50$, `expanded_bench.py`). The shim service time is the appropriate input to the queue model below.

Queue dynamics at $\lambda = 13.5$ TPS, $c = 2$. Measured $\mu_{\text{SLH-DSA-SHA2-128s}} = 3.777$ ops/s per worker (mean across settled trials; derived from shim service time), so $\rho = 13.5 / (2 \times 3.777) = 1.787$ (95% CI: [1.759, 1.816]; $N = 8$) > 1 . Queue depth grows at $\lambda - c\mu = 13.5 - 7.55 = 5.95$ ops/s. Queue depth: ~ 357 ops at $T+60$ s (95% CI: [349, 364] ops); ~ 1785 ops at $T+300$ s. ρ first exceeds 1 immediately on algorithm flip ($T+0.0$ s; follows directly from $\lambda = 13.5 > c\mu_{\text{SLH-DSA-SHA2-128s}} = 7.55$).

Realism

All four conditions of Section 3.5 are satisfied: SLH-DSA is in the supported algorithm set by configuration (condition 1); the A1 adversary has configuration write access by assumption (condition 2); baseline $\rho \ll 1$ provides headroom (condition 3); per-operation cost 261,945 μ s (isolated) dominates per-trigger attacker cost (~ 100 μ s for the configuration PUT; condition 4). The headline $\text{CAF}_{r_{\text{cpu}}}$ for this case (V_{NPP}) is in Table 1.

5.2. A3: Remote TLS Peer Against OQS-OpenSSL 3

Setup

The `vulnerable_server` testbed node runs OQS-OpenSSL 3 via the `openquantumsafe/oqs-openssl3` Docker image (oqs-provider 0.11.0, OpenSSL 3.6.0; see Section 2.3). The server's cert chain contains three leaf certificates sharing a self-signed root:

ECDSA-P256, ML-DSA-65, and SLH-DSA-SHA2-128s. The TLS 1.3 server's `-sigalgs` flag advertises all three in priority order `ecdsa_secp256r1_sha256:mldsa65:slhdsa_sha2_128s`. *Measurement platform* TLS handshake timing is collected inside the `openquantumsafe/oqs-oss13` container (oqs-provider 0.11.0, OpenSSL 3.6.0, Linux/ARM64, Docker loopback). On macOS with Homebrew OpenSSL 3.6.2, oqs-provider 0.8.0 exhibited an algorithm-registration incompatibility that prevented TLS server startup; oqs-provider 0.11.0 via Docker is the compatible alternative. The per-sign cost is cross-validated against the A1 isolated Python benchmark ($261,945\ \mu\text{s} \approx 262\ \text{ms}$; $n = 50$; expanded bench 2026-04-18), confirming consistent library behaviour across platforms.

This configuration mirrors a plausible BSI/ANSSI/NCSC-aligned high-assurance deployment that follows PQ-migration guidance and includes SLH-DSA in the certificate chain. The server is purpose-built for this experiment; no public TLS deployment currently carries SLH-DSA certificates (confirmed by the census of Section 6.1). The experiment demonstrates the *mechanism*, not current prevalence: production exposure begins when certificate issuance starts, which the BIS/G7 roadmaps and FIPS 205 finalisation make foreseeable within this decade.

Exploitable property

The finding is *not* “SLH-DSA is slow” (known) nor “the operator’s configured algorithm costs what it costs” (a tautology). It is a *consent asymmetry*. An operator may provision SLH-DSA for a narrow, legitimate reason: an offline root CA, a constrained-client niche, a FIPS 205 compliance obligation. Crucially, the exposure does not require a single negligent act; it arises from the *composition* of individually defensible decisions. A BSI/ANSSI/NCSC “support all NIST families” mandate places SLH-DSA in the supported set (Section 8.6); a LAMPS SLH-DSA X.509 or PQT dual-certificate deployment [29,44] places an SLH-DSA leaf in a *negotiated* chain by design; and shared HSM/CA infrastructure serving both an offline root-signing function and an online mTLS terminator—routine in cost-constrained payment estates—brings that leaf within the online selection surface. Each step is standards-conformant, yet their conjunction hands an unauthenticated peer an online lever over a $\sim 10^4 \times$ cost asymmetry: the fix belongs at the protocol and policy layer (Sections 7.1 and 7.2, PSA), since there is no single misconfiguration to avoid. Nevertheless, on the *shared online* endpoint, TLS 1.3 hands any unauthenticated peer a unilateral, zero-cost selector (one ClientHello, a few hundred bytes) over a $\sim 10^4 \times$ server-cost lever, with no protocol mechanism to separate the legitimate use from the adversarial one. TLS 1.3 is operating as specified: the server selects from the intersection of the client’s `signature_algorithms` and its own set—but provides no means (nor does any IETF draft we surveyed) to weight selection by CPU cost, rate-limit high-CAF choices, or decline a ClientHello that would push ρ past the boundary. The server pays 262 ms per SLH-DSA connection; the attacker pays one ClientHello. What is new here is this unauthenticated, zero-cost, unilateral selector over an operator-provisioned cost lever, not the latency of any one algorithm.

Attack chain

The attack chain is as follows:

1. The attack client (Go) opens a TCP connection to the vulnerable server.
2. Sends a TLS 1.3 ClientHello whose `signature_algorithms` extension lists only `slhdsa_sha2_128s`. The server’s intersection contains exactly one algorithm: SLH-DSA-SHA2-128s.
3. The server selects the SLH-DSA leaf and computes CertificateVerify, costing 262 ms of server CPU. No error condition is raised; the server is doing exactly what the TLS specification requires.

4. The attack client receives ServerHello and Certificate, then closes the connection. No application data is exchanged. The server has already spent the 262 ms.

The load driver supports a rate sweep $r \in \{1, 10, 100, 1000\}$ handshakes per second, each for 30 s, from a single origin and then from a second origin in a different cloud region; the $r = 10$ trace reported below is the representative saturation run.

Measurement

Three channels: (i) server-side `rusage` per worker thread, extracted from `node_exporter` and from a patched `oqs-provider` that writes per-sign wall time to a text log; (ii) connection backlog polled via `ss -tln` at 100 ms resolution; (iii) handshake completion time at the attack client, measured via Go's `tls.Dialer` instrumentation.

Results

Table 3 reports TLS handshake wall-clock timing measured inside the `openquantumsafe/oqs-oss13` container (oqs-provider 0.11.0, OpenSSL 3.6.0, Linux/ARM64 Docker loopback, $N = 20$ per algorithm, `BusyBox time` 10 ms resolution).

Table 3. A3 TLS handshake wall-clock time (server signs each connection). SLH-DSA handshake is dominated by the signing operation (~ 262 ms of 280 ms total); ECDSA handshake is dominated by protocol overhead ($\ll 1$ ms signing of 30 ms total). p95/p99 collapse to the same bucket (e.g., ECDSA p95 = p99 = 40 ms) due to `BusyBox time` 10 ms resolution; p95/p99 distinctions are not meaningful at this resolution.

Algorithm	n	p50 (ms)	p95 (ms)	p99 (ms)
ECDSA-P256	20	30	40	40
SLH-DSA-SHA2-128s	20	280	280	280
Δ (attack cost)		+250 ms per connection		

The SLH-DSA handshake wall time (280 ms) is consistent with the isolated per-sign benchmark (262 ms, Section 5.1): signing accounts for $\sim 94\%$ of the full TLS round-trip. The ECDSA handshake (30 ms) is dominated by protocol and process overhead; its signing component (21 μ s) is negligible. The TLS-level wall-clock ratio is $280/30 = 9.3\times$; the CPU-level $\text{CAF}_{r_{\text{cpu}}}$ (Table 1) is the correct metric for queue-saturation analysis, because the queue is bounded by the *CPU cycles consumed per operation*, not by wall-clock round-trip time.

The server-side saturation analysis follows the same queue model as A1 with c replaced by the number of TLS worker threads. For a single-core configuration ($c = 1$), the wall-clock per-handshake time of 280 ms (Table 3) yields a *client-observed* rate of $1/280 \text{ ms} = 3.57 \text{ handshakes/s}$ and a predicted $\rho = r/3.57$ that crosses 1 at $r \approx 3.6/\text{s}$ (so the smallest integer attack rate that saturates is $r = 4/\text{s}$). The wall-clock value is conservative for queue analysis because it includes client-side TCP/TLS round-trip components that do not block server throughput; the actual server-side service rate during sustained load is bounded below by the per-sign CPU cost ($\sim 262 \text{ ms isolated} \Rightarrow \sim 3.8 \text{ signs/s}$ on a single core); under the sustained flood below, it was measured at $\sim 7.3 \text{ completions/s}$ ($\approx 137 \text{ ms effective per handshake}$ —faster than the isolated figure, owing to warm caches and pipelined I/O overlap, with CPU pinned near one core). Either estimate gives $\rho > 1$ at $r = 10/\text{s}$ and unbounded queue growth. From the attacker's perspective, each connection requires only sending a ClientHello with `signature_algorithms = [slhdsa_sha2_128s]`; the attacker waits 280 ms for the full round-trip, or can pipeline multiple concurrent connections to saturate the server faster.

Load test at $r = 10 \text{ HPS}$. We dispatched $r = 10$ SLH-DSA ClientHellos per second for 30 s (300 total connections) against the persistent `oqs-oss13` server (`naccept=100`), using the

shell load driver. By the end of the window (extended past the 30 s dispatch to allow drain), 219 connections completed the handshake and 81 were rejected once the accept queue exhausted its naccept budget. The ~ 7.3 completions/s imply a server-side service rate of ~ 137 ms/handshake (below the 280 ms client wall-clock, which includes non-blocking client/network components); container CPU peaked at 103.5% of one core, confirming saturation. A concurrent benign ECDSA stream ($\lambda_{\text{benign}} \approx 0.5/\text{s}$ on a separate port) saw p99 rise from 76 to 91 ms (+20%)—*illustrative only*, since attacker, victim, and benign client are co-resident and the elevation reflects shared-scheduler contention, not a distributed estimate. At the observed rate $\rho = 10/7.3 \approx 1.37 > 1$ (or ≈ 2.80 using the conservative wall-clock rate), both exceed the stability boundary, and the server was observably saturated (CPU pinned, 81 rejections). Raw results are in the artefact.

Realism. All four conditions of Section 3.5 are satisfied: SLH-DSA is in the server’s configured cert chain (condition 1); the A3 adversary is any TLS client, requiring no credentials or insider access (condition 2); baseline ECDSA $\rho \ll 1$ provides headroom (condition 3); per-operation cost ≈ 262 ms dominates attacker’s ClientHello cost $\ll 0.1$ ms (condition 4). The headline $\text{CAF}_{r_{\text{cpu}}}$ for this case is in Table 1.

5.3. A2: Supply-Chain (Modelled)

A2 shares the amplification primitive of the two demonstrated attacks (A1 and A3)—adversarial algorithm selection—and differs from them only in the delivery channel and its reach. We therefore measure its amplification on a controlled build and model only the distribution-channel compromise, which we cannot ethically exercise against live third-party infrastructure (detailed below). We model A2 on a controlled liboqs build (pinned 0.14.0) patched with a preference-shim that silently promotes SLH-DSA in the OQS algorithm ordering—a single function override in `liboqs/src/sig/sig.c`, installed in a sandboxed chroot, never ingested by any production system. Analytically, a victim that upgrades the compromised library inherits the modified ordering, so the next handshake whose cert chain offers both ECDSA and SLH-DSA selects SLH-DSA, provided the client advertises it (the default in OQS-OpenSSL client builds). We do not model the upstream-compromise mechanism (a signing-key or build-pipeline intrusion).

Exploit chain. Concretely, the injection has five stages. (i) An upstream artefact is compromised—a build-pipeline or package-registry intrusion, or a poisoned container base image (modelled, not performed; see below). (ii) The dispatch logic in `liboqs/src/sig/sig.c` is patched so the algorithm enable/ordering path promotes an SLH-DSA identifier ahead of the ECDSA/ML-DSA entries returned to the caller—a localised change to the `OQS_SIG` table that alters no function signature and leaves the build green. (iii) The patched library is published under the expected version, or a permissive version range, defeating naive pinning (exercised on the controlled build described below, not published to any live channel). (iv) The victim’s CI rebuilds oqs-provider against it and deploys. (v) At runtime, every handshake whose negotiated intersection admits SLH-DSA is signed with SLH-DSA, so the measured per-operation $\text{CAF}_{r_{\text{cpu}}} = 12,255\times$ and the $\rho > 1$ saturation established for A1 and A3 apply unchanged.

Why we withhold a live proof-of-concept. Stages (ii)–(v) are exercised on the controlled backdoored-preference build above and are empirically grounded in the same measured cost as A1 and A3; only stage (i)—a genuine compromise of a real distribution channel—is modelled rather than carried out. The controlled build is instrumented to log the negotiated CertificateVerify scheme on each handshake; with the `liboqs/src/sig/sig.c` preference shim installed, an otherwise-identical handshake selects SLH-DSA-SHA2-128s in place of ECDSA-P256—the same selection flip driven directly in the A1 testbed (Section 5.1)—and the resulting per-sign cost is the measured 262 ms of Table 1, so the

code-modification effect is observed rather than assumed. We withhold the live channel-compromise step because performing it would attack live third-party infrastructure and cross the ethical and legal boundary we set in Section 4.8. The amplification, not the intrusion, is this paper's contribution, and it is measured rather than assumed.

The per-sign CAF under A2 equals A3's—algorithm selection is the amplification primitive, not the delivery mechanism—and the victim service rate μ and stability condition $\rho > 1$ are identical, since it is the same algorithm on the same hardware. What A2 adds is *reach*: one library compromise propagates to every deployment that upgrades, with no per-victim network presence, so its amortised per-trigger cost approaches zero. Realism conditions (3) and (4) follow from the A3 measurements; (1) and (2) hold without per-victim privileges—the class's distinguishing property—and are broken by reproducible builds and library-signature verification.

5.4. Cross-Deployment Summary

The pattern is uniform across A1–A3: all three produce the same headline $\text{CAF}_{r_{\text{cpu}}}$ (Table 1) and cross the queue-stability boundary at realistic arrival rates—A1 at $\rho = 1.787$ ($\lambda = 13.5$ TPS, $c = 2$), A3 at $r \geq 4$ handshakes/s, A2 on every connection of an affected deployment. The attack does not require volume; it requires presence in the negotiation.

6. Threat Verification

This section provides empirical verification that the threat described in Section 4 is real and operationally relevant, through two independent lines of evidence: (i) an internet-scale passive TLS scan confirming the current deployment state, and (ii) a record of the responsible-disclosure notifications we sent to the affected maintainers and the IETF TLS working group.

6.1. Internet-Scale Deployment Census

To bound the current in-scope surface empirically, we conducted a non-exploiting TLS handshake census against the Tranco top-1000 domains (fetched 18 April 2026), probing port 443 with a standard `ssl.SSLContext` handshake and recording only the certificate signature algorithm OID and TLS version. No application data was exchanged; the probe is equivalent to a browser connecting to the site. We additionally scanned a curated list of 35 high-value targets (cloud providers, national CSIRT agencies, financial institutions, Australian critical-infrastructure operators).

Results are as follows:

- **Tranco top-1000:** 1000 targets probed; 692 successful TLS handshakes (308 failures: DNS, timeout, or certificate errors). Observed certificate signature algorithms: `sha256WithRSAEncryption` (488), `ecdsa-with-SHA256` (99), `ecdsa-with-SHA384` (87), `sha384WithRSAEncryption` (18). *PQC signature algorithms detected: 0 (0.00%).*
- **Curated high-value list:** 35 targets (Cloudflare, Google, Microsoft, AWS, Apple, NIST, AusCERT, NCSC, major Australian banks); 32 successful connections. *PQC signature algorithms detected: 0.*

Interpretation

The scan confirms no PQ-signature deployment in public TLS as of April 2026, consistent with Cloudflare's "not a single public post-quantum certificate" [1]—validating the paper's pre-deployment framing: the threat is latent in standards and reference implementations but not yet in production, so the window to deploy the defences in Section 7 before migration remains open. The census measures *prevalence*, which is zero; it does not measure *feasibility*, which is already non-zero. The components of a vulnerable

deployment ship today: Entrust nShield v13.8.0 is a CAVP-validated HSM that natively signs SLH-DSA [26] (the signing half), and the LAMPS SLH-DSA-X.509 and PQT dual-certificate drafts [29,44] standardise SLH-DSA leaves in negotiated chains (the certificate half). What is pending is only their *composition* into a production-reachable endpoint—an integration step, not a research advance—which is precisely why the correct intervention point is now, before that composition hardens into long-lived certificates. A zero-prevalence census is therefore the expected reading of a pre-deployment analysis, not a refutation of it. Methodologically, the probe is equivalent to a browser reading each certificate—no exploitation, credential capture, or application-layer interaction—bounded by an 8-second timeout and 80 concurrent workers over publicly reachable services; raw data is in the artefact.

6.2. Responsible Disclosure

The attack exploits *intended* TLS 1.3 behaviour, so there is no protocol-level CVE, and the implementation observations we report concern publicly visible default configurations rather than latent defects. Prior to publication, we nonetheless notified the affected maintainers and the relevant standards body. We notified the OQS project that oqs-provider's default advertises SLH-DSA in `signature_algorithms` with no selection enforcement or rate limiting—an out-of-the-box A3 exposure—recommending optional selection-enforcement and rate-limit hooks and a high-CAF warning. The OQS maintainers acknowledged the report, noted that the stack targets research and experimentation rather than production deployment, and welcomed a documentation contribution making that scope explicit. We notified Entrust that nShield exposes SLH-DSA with no PKCS#11 algorithm-tier policy, recommending a policy template that caps high-cost algorithms to a bounded-capacity pool. We also raised the issue with the IETF TLS working group, proposing a security-considerations note for the ML-DSA-in-TLS [11], SLH-DSA X.509 [29], and PQT dual-certificate [44] work that would constrain high-CAF algorithms to per-tier capacity pools. At the time of writing, the Entrust and IETF responses are pending.

We do not release the attack proof-of-concept publicly; it is shared only with verified researchers on request (see the Data Availability statement), to avoid facilitating misuse.

7. Defences

The four defences below are layers of a single defence-in-depth stack, not a pick-one menu: the hard admission controls D1 and D2 carry the formal $\rho_{\text{SLH-DSA}} \leq \alpha < 1$ guarantee, D3 adds early warning, and the Parallel Signing Architecture combines all three into one deployable architecture (Proposition 1). This section describes three defensive primitives (D1 algorithm—selection enforcement, D2 algorithm—keyed rate limiting, D3 CAF—spike anomaly detection) and a composite architectural defence (the Parallel Signing Architecture, PSA). Each primitive operates at a different layer so the reader can choose based on deployment constraints. We measure each under the A1 and A3 attack traces from Section 5 and report the ρ -bound each defence achieves, legitimate-traffic overhead, detection latency, and deployment cost.

7.1. D1: Algorithm-Selection Enforcement Design

A network-layer filter (a TLS-terminating proxy or a sidecar) inspects each incoming ClientHello's `signature_algorithms` extension. If the intersection with the server's cert chain would select an algorithm flagged as *high-CAF for the destination service tier*, the connection is either rejected with TLS alert `unsupported_certificate` (strict mode) or steered to an isolated low-throughput service tier (graceful mode).

Per-tier policy is a mapping $\text{tier} \rightarrow \mathcal{S}_{\text{allowed}}$; the high-throughput tier excludes any algorithm with measured $\text{CAF}_{\text{cpu}} \geq 100\times$ relative to the tier's baseline. The $100\times$ default lies within the natural gap between the most expensive non-SLH-DSA variant (FN-DSA-1024 at $\leq 14.3\times$ across all tested architectures; Table A1) and the cheapest SLH-DSA variant (SLH-DSA-SHA2-128f at $\geq 192\times$), cleanly separating hash-based from lattice/Falcon algorithms. The threshold is operator-configurable; Section 3 provides the canonical table deployments use to populate it.

Implementation and evaluation

A reference filter (Envoy plus a standalone Go proxy) reads the ClientHello before the TLS session is established, adding no crypto cost on the allowed path. Against a synthetic trace of 90 SLH-DSA requests at 3 TPS, strict mode rejected 100% before any signing ($\rho_{\text{SLH-DSA}} = 0$); benign ECDSA was unaffected (p50 437 μs , p99 576 μs , versus 455/566 unprotected). Graceful mode steers SLH-DSA to the isolated tier, capping its arrival rate at $\alpha\mu(A)$ so $\rho_{\text{SLH-DSA}} \leq \alpha < 1$.

Adaptive adversary

An attacker who knows D1 is deployed can include other algorithms in the ClientHello so the intersection selects ML-DSA/ECDSA—graceful degradation that leaves at most $14.3\times$ (FN-DSA-1024) or $4.8\times$ (ML-DSA-65) (cross-architecture maxima across all tested microarchitectures; the ARM64 testbed values are $10.7\times$ and $2.8\times$), far below the saturation threshold (Table A1)—or attack via Encrypted ClientHello [45] (Section 7.7). D1 is thus resilient for the queue-saturation goal, though not for every DoS goal.

7.2. D2: Algorithm-Keyed Rate Limiting

Design

A token-bucket rate limiter keyed by (algorithm, op_type) rather than by client identity. The token rate for SLH-DSA-SHA2-128s is set to $c \cdot \mu(A) \cdot \alpha$, where $\alpha \leq 0.5$ is a safety factor preserving queue headroom. ML-DSA-65 gets a much larger budget; ECDSA-P256 is unrestricted (or limited by a per-client policy only).

Implementation

A sidecar intercepts the CertificateVerify path with per-service token-bucket state. Reference rates ($c = 2$, $\alpha = 0.25$, ARM64): ≤ 1.858 SLH-DSA signs/s ($2 \times 3.717 \times 0.25$; μ_{D2} below) and ≤ 8278 ML-DSA-65 signs/s; ECDSA unlimited.

Evaluation

The token budget caps the admitted SLH-DSA arrival rate at $\lambda_{\text{cap}} = c \cdot \mu(A) \cdot \alpha = 2 \times 3.717 \times 0.25 = 1.858$ ops/s. (Note on μ source: D2 uses $\mu_{\text{D2}} = 3.717$ ops/s, derived from the shim service time at the 95th-percentile CI upper bound ($10^6/269,066 \mu\text{s} = 3.717$ ops/s; A1 measurements); the A1 analysis uses the empirical mean $\mu_{\text{A1}} = 3.777$ ops/s (264,804 μs). The D2 bound uses the conservative upper-shim value to avoid over-provisioning the token budget.) The maximum-achievable queue utilisation under D2 is therefore $\rho_{\text{max}} = \lambda_{\text{cap}} / (c \cdot \mu(A)) = \alpha = 0.25 < 1$: the queue *cannot* saturate while D2 is active, by construction (the token rate is set as a fraction of capacity). This is an algebraic bound, not an empirically derived one.

Empirical evidence (at sub-saturation $\rho < 1$ on this testbed; the genuine-saturation caveat is stated in Section 7.6) comes from the testbed evaluation (A1 trace, $\lambda_{\text{atk}} = 3$ TPS, token rate 1.858 ops/s): D2 rejected 31.1% of attack requests (theoretical steady-state: $(3 - 1.858)/3 = 38.1\%$; lower measured rate due to token-bucket burst capacity during the finite-duration 90-request test window), keeping measured $\rho = 0.40 < 1$. The measured ρ

exceeds $\alpha = 0.25$ because the end-to-end service time during D2 operation exceeds the shim benchmark μ_{D2} used to set the token rate (runtime D2-sidecar overhead effectively reduces μ , raising the realised ρ). The queue remains stable ($\rho < 1$) in all cases. Legitimate-traffic impact on the ECDSA path: zero (unmetered by D2).

Tail-latency impact

We measure p99 of handshake completion under near-cap SLH-DSA load ($\lambda_{\text{atk}} = 3$ TPS, token rate 1.858 ops/s). Benign ECDSA p99 under D2: 641 μs versus 566 μs unprotected, a $1.13\times$ overhead. The SLH-DSA path itself is throttled to ≤ 1.858 ops/s; legitimate bursty archival-signing workloads must queue or be deferred. The full benign p99 CDF across all three defence configurations is available in the reproducibility package (available on request).

7.3. D3: CAF-Spike Anomaly Detection

Design

A streaming statistical monitor over per-second service-level utilisation ρ at the cryptographic-operation queue. Two thresholds are used:

- $\rho > 0.5$ sustained for ≥ 5 s: alert; begin load-shedding low-priority traffic.
- $\rho > 0.8$ sustained for ≥ 2 s: full circuit-breaker; reject new high-CAF requests with TLS alert `internal_error`.

A second monitor tracks the *algorithm-mix* as a rolling z-score: a sudden spike in SLH-DSA selection rate ($z > 4$ over a 60-s window) raises alert independent of ρ , detecting the attack *before* queue dynamics catch up.

Implementation

The monitor is implemented as a Prometheus alerting-rule pair plus a small Go control-plane component that toggles D2's token rates via SIGHUP. The circuit-breaker action is the same rejection primitive used by D1 in strict mode.

Evaluation

We drove the shim with ECDSA at 13.5 TPS for 5 s, then flipped to SLH-DSA-SHA2-128s at $T = 0$; D3 polled `/metrics` every 0.5 s with an EMA ($\tau = 5$ s) over five trials (Table A2). The ρ -alert fired in 4/5 trials (median 18.5 s); the z-score monitor in 3/5 (median 21.3 s, high variance as the 60 s window absorbs the attack into its own baseline); the circuit-break ($\rho > 0.8$) never fired. All three exceed the analytic bounds (≤ 7 , ≤ 5 , ≤ 3 s) because the measurement host out-paced the assumed $\mu = 3.777$ ops/s, so ρ_{true} settled near 0.6 rather than 1.787; the bounds hold once $\rho_{\text{true}} > 1$ on real hardware.

Calibration requirement. D3 thresholds must be calibrated to the operator's actual hardware μ , not to benchmark values: run a 60 s isolated SLH-DSA sign loop on the target hardware and set $\mu_{\text{calib}} = 1/\bar{t}_{\text{sign}}$. Without calibration, the circuit-break may never fire on fast hardware, as observed here. The EMA step-response derivation of the analytic bounds (≤ 7 s ρ -alert, ≤ 5 s circuit-break, ≤ 3 s z-score) and the full detection-latency table are given in Appendix B.

Adversary adaptation

D3 is not adversarially robust against a low-and-slow attacker who paces SLH-DSA requests to keep both ρ below 0.5 and the algorithm-mix z-score below 4 (both D3 thresholds must be evaded simultaneously). We state this as a limitation; the adaptive detectors described next mitigate, but do not eliminate, it.

Adaptive thresholds

The static triggers above ($\rho > 0.5, z > 4$) are deliberately simple; three online-adaptive replacements reduce false positives under diurnal load and remove the manual-calibration burden. (i) Self-calibrating utilisation: estimate μ continuously from completed-signature timings with a P-square online quantile estimator, so the ρ threshold tracks the hardware without the 60 s calibration loop above. (ii) Change-point detection: replace the fixed $z > 4$ rule with a CUSUM or Page–Hinkley test on the SLH-DSA selection rate, which signals a sustained distribution shift faster than a fixed-window z-score and prevents the 60 s window from absorbing the attack into its own baseline. (iii) Trend test: a Mann–Kendall test on the ρ trajectory flags a monotone climb before either fixed threshold is crossed. The trade-off is the poisoning risk noted above—a slow-adapting baseline can be dragged upward by a low-and-slow attacker—so an adaptive detector must bound its adaptation rate. Crucially, availability does not depend on D3: the D2 token-bucket cap of Proposition 1 admits SLH-DSA at rate at most $\alpha c' \mu_{\text{SLH-DSA}}$, irrespective of the detector’s state, so poisoning the D3 baseline cannot lift the admission cap or the $\rho_{\text{SLH-DSA}} \leq \alpha < 1$ guarantee. Clamping the self-calibrating μ estimator from above at the calibrated μ_{calib} further ensures an attacker cannot inflate the estimated service rate to shrink the reported $\rho = \lambda / (c\mu)$, so the adaptive trigger can only tighten, never relax, relative to the static rule. We prototype the change-point variant but leave a full evaluation to future work.

7.4. PSA: Parallel Signing Architecture (Composite)

The Parallel Signing Architecture (PSA) structurally combines the above: a dedicated low-throughput SLH-DSA lane with capacity $c' \ll c$ (often $c' = 1$) is provisioned alongside the main ECDSA / ML-DSA lane. Token-bucket limits on the SLH-DSA lane are D2; the strict algorithm-to-lane mapping at ingress is D1 in graceful mode; the anomaly detector (D3) sits across both lanes. PSA’s effectiveness in a payment-systems HSM context is further studied in [16].

PSA bounds the SLH-DSA utilisation on both lanes by admission control, for any adversarial arrival rate λ_{atk} . We state the guarantee formally.

Proposition 1 (PSA queue stability). *Let SLH-DSA signing requests arrive at an arbitrary (possibly adversarial) rate λ_{atk} and be served by c' dedicated lanes of service rate $\mu_{\text{SLH-DSA}}$ each. Assume the adversary controls the arrival times, volume, and advertised signature-algorithm selection of SLH-DSA requests but can neither forge admission tokens nor reach a signing lane except through the algorithm-keyed bucket. If PSA admits SLH-DSA requests through an algorithm-keyed token bucket at rate at most $\lambda_{\text{cap}} = \alpha c' \mu_{\text{SLH-DSA}}$ for a fixed $\alpha \in (0, 1)$, then the SLH-DSA signing subsystem is stable, with utilisation $\rho_{\text{SLH-DSA}}^{\text{iso}} \leq \alpha < 1$ and finite stationary mean queue length and waiting time, bounded independently of λ_{atk} .*

Proof. D1 graceful-mode routing admits no SLH-DSA request to the main ECDSA/ML-DSA lane, so $\rho_{\text{SLH-DSA}}^{\text{main}} = 0$. On the isolated lane, the admitted rate obeys $\lambda_{\text{adm}} \leq \lambda_{\text{cap}} = \alpha c' \mu_{\text{SLH-DSA}}$ by the token-bucket constraint, whatever the offered $\lambda_{\text{atk}} \geq \lambda_{\text{adm}}$ (excess is shed or queued at admission, not at the signing lanes). Concretely, a token bucket of rate λ_{cap} and depth B admits at most $\lambda_{\text{cap}} t + B$ requests in any window of length t , so the long-run admitted rate obeys $\lambda_{\text{adm}} \leq \lambda_{\text{cap}}$ regardless of λ_{atk} ; the finite depth B adds only a bounded transient and does not affect the utilisation. Hence, $\rho_{\text{SLH-DSA}}^{\text{iso}} = \lambda_{\text{adm}} / (c' \mu_{\text{SLH-DSA}}) \leq \alpha < 1$. For an M/M/ c' lane with $\rho < 1$, the Erlang-C mean wait $W_q = C(c', \rho) / (c' \mu_{\text{SLH-DSA}} - \lambda_{\text{adm}})$ is finite; for general service distributions—including the near-deterministic SLH-DSA case (Section 2.6)— $\rho < 1$ gives a finite stationary mean wait by Loynes’ criterion [33]. Both quantities are independent of λ_{atk} because admission caps the lane input. $\square$

By contrast, the undefended shared pool has $\rho = 1.787 > 1$ (Section 2.5), so its queue grows without bound; Proposition 1 is precisely the property that restores stability. The per-operation $\text{CAF}_{r_{\text{cpu}}}$ is unchanged by PSA; what the architecture removes is the attacker's ability to drive $\rho > 1$ on either lane.

Scope of the formal claim

The property that Proposition 1 establishes is an *availability* property—queue stability under adversarial load—for which the appropriate formalism is queueing theory rather than symbolic protocol analysis. Secrecy and authentication of the underlying hybrid TLS 1.3 handshake are orthogonal and rest on the protocol's existing analyses; mechanised cryptographic verification (e.g. Tamarin or ProVerif models of the handshake) is complementary and outside the scope of this availability-focused study. The boundary is principled, not merely one of convenience: symbolic (Dolev–Yao) models abstract each cryptographic operation as an atomic, unit-cost term with no notion of per-operation CPU time or queue occupancy, so a cost-asymmetry availability property is not expressible in them; the service-rate-versus-arrival-rate quantity this attack manipulates lives natively in the queueing formalism of Proposition 1.

Deployment cost

One additional service-tier's worth of compute capacity; typically, a single additional HSM slot or software-crypto worker pool. For the NPP-representative testbed, $c' = 1$ on one extra SoftHSMv2 slot; incremental cost is negligible.

Scaling to large clusters

The stability guarantee of Proposition 1 is *per-lane* and therefore scale-free: it constrains each lane's utilisation through admission control, not aggregate throughput, so it holds for any number of signing lanes or HSM slots. For an N -slot SLH-DSA cluster, algorithm-keyed admission is enforced at cluster ingress—either a shared token bucket capping the aggregate admitted SLH-DSA rate at $\alpha N \mu_{\text{SLH-DSA}}$ ($\alpha < 1$), or, to avoid a central choke point at large N , that same budget partitioned across the k ingress nodes into local buckets of rate $\alpha N \mu_{\text{SLH-DSA}} / k$ each, requiring no per-request cross-node coordination; the local rates sum to the aggregate cap and each admits no more than the shared bucket would, so every lane still inherits $\rho_{\text{SLH-DSA}}^{\text{iso}} \leq \alpha < 1$ —while requests are spread across slots by a load-balancing rule that keeps the busiest lane close to the mean: join-shortest-queue, or the lighter power-of- d -choices ($d = 2$) probe, both of which hold the maximum-to-mean utilisation gap small even as N grows. The admission cap then makes every lane inherit $\rho_{\text{SLH-DSA}}^{\text{iso}} \leq \alpha < 1$, so the two-slot result extends to cluster scale without re-derivation. Two engineering caveats remain for future work: the cross-slot coordination overhead of shortest-queue routing at high N , and slot heterogeneity (clusters mixing ML-DSA- and SLH-DSA-capable slots) that complicates the routing decision.

7.5. Defence Evaluation Harness

Our defence-evaluation harness replays a recorded attack trace against the unprotected and protected service in turn, measuring pre/post ρ trajectories, legitimate-traffic latency CDF, detection latency, and circuit-breaker recovery time. The output is a single CSV used to produce the defence-efficacy CDF plot.

7.6. What We Do Not Claim

- The defences do not address the *existence* of high-CAF algorithms. They mitigate adversarial exploitation of a deployment's negotiation surface.

- D3's anomaly detection is not adversarially robust in the strong sense. This will be future work.
- D1 relies on visibility into ClientHello; encrypted ClientHello is a deployment constraint that requires ECH termination at the ingress.
- None of these defences obviate the need for configuration hygiene (algorithm-tier policies) at HSM and certificate-authority layers; they are a defence-in-depth layer, not a replacement.
- We do not claim empirical defence validation under genuine queue saturation. The D1/D2/PSA $\rho \leq \alpha < 1$ bounds are immediate algebraic consequences of token-bucket admission control (Proposition 1); the empirical contribution is the deployment design and the measured overhead/latency. The defensive novelty is not the token bucket but its *key*: classical TLS-DoS rate-limiting keys on client identity, whereas here the scarce resource is keyed by the *algorithm the unauthenticated peer selects* (Section 7.2), a keying that becomes meaningful only once the supported set spans an order-of-magnitude signing-cost range—the defining property of hybrid post-quantum negotiation. Because the testbed's signing throughput exceeded the benchmark-derived μ , the live defence runs operated at $\rho \approx 0.4\text{--}0.6$, *not* the $\rho_{\text{true}} = 1.787$ regime the defences are designed for. The sub-saturation runs are not merely a shortfall; they confirm the admission *mechanism* operates as modelled: D2 measurably rejected 31.1% of attack requests and held $\rho = 0.40 < 1$ while leaving the ECDSA path unmetered (Section 7.2). Because a token bucket's admitted rate is capped independently of the offered rate (Proposition 1), the same validated mechanism yields $\rho \leq \alpha$ at any higher λ_{atk} ; what the live runs leave unexercised is the operating point, not the admission logic. Efficacy under genuine saturation ($\rho > 1$) is thus established analytically only. Hardware-rate-matched end-to-end defence evaluation, and larger sustained-flood traces than the 90-request/300-connection calibration sets used here, are future work (Section 10).

7.7. Defence Deployment Trade-Offs

The defences are not free, and their limits motivate the attack's design: an adversary targeting a deployment that has not yet adopted these controls faces no algorithmic barrier. Three trade-offs bound their reach. D3's detection latency is environment-dependent: on a compute-rich testbed whose service rate exceeds the modelled μ , the queue never enters the $\rho > 1$ regime the alarms are tuned for, so the circuit-break may not fire and ρ -alert latency stretches well past its analytic bound (Section 7.3). D1 needs visibility into the client's `signature_algorithms` extension, which Encrypted ClientHello removes unless the deployment terminates and re-inspects at a controlled edge—the configuration typical of the BSI/ANSSI/NCSC-aligned deployments we target (Section 7.1). And no fixed-threshold detector is robust against a low-and-slow attacker pacing requests below both alarms; there, the hard admission controls D1 and D2 remain effective (Section 7.2).

8. Scope and Threat-Surface Bounds

This section makes explicit which deployments are exposed to the attack we describe, which are not, and which become exposed as planned PQ-signature rollouts proceed. The attack is an *emerging deployment risk*, not a broadly deployed threat: currently latent in production but foreseeable in deployment planning. The paper's credibility depends on being precise about which deployments fall on which side of that line.

8.1. In Scope Today

The attack applies to any deployment whose TLS 1.3 cert chain (or analogous federated-identity construction) includes SLH-DSA, or whose HSM configuration permits SLH-DSA

signing on shared capacity with ECDSA/ML-DSA workloads. Concretely, the following are in scope:

- BSI/ANSSI/NCSC high-assurance contexts. German, French, and UK guidance includes SLH-DSA in approved long-term signature sets [46–48], so financial, telecom, and critical-infrastructure deployments in these jurisdictions will carry it. (NSA CNSA 2.0 differs: SLH-DSA is excluded from general NSS use, permitted only for software/firmware signing [49].)
- Software and firmware signing globally—the use case CNSA 2.0 permits and the canonical target for hash-based signatures. Signing infrastructure (CA, code-signing KMS, firmware-update PKI) exposing an algorithm-selecting endpoint is directly vulnerable to A1/A2.
- Financial-sector PQ migration. BIS Papers No. 158 targets late-2020s migration [50] and the G7 CEG roadmap 2030–2032 [51]; RTGS, instant payments, SWIFT, and interbank mTLS traverse high-assurance infrastructure in this window. The Australian NPP (Section 5.1, [16]) is representative.
- IETF-draft hybrid-signature deployments. The composite-signature, SLH-DSA X.509, and PQT dual-cert drafts [28,29,44] specify the machinery for SLH-DSA chains; reference deployments are the cleanest A3 targets.
- Misconfigurations. Any HSM whose firmware supports SLH-DSA without a per-tier allowlist risks an A1 realisation; Entrust nShield v13.8.0 is the only HSM with confirmed native SLH-DSA in our sweep [26], but this is changing as vendors ship PQ firmware.

In each case, the exposure is *emerging* rather than current: it becomes operationally relevant as SLH-DSA certificates and HSM signing move from pilot to production over the FIPS 205 rollout window, a transition our Internet census (Section 6.1) shows has barely begun (0.00% of surveyed public endpoints today). The deployments above—high-assurance government and defence TLS, real-time payment and interbank signing, and SLH-DSA-issuing PKI/CA infrastructure—are precisely where that transition lands first, and therefore where the attack is anticipated to matter earliest.

Concrete scenario

As a representative early-exposure case, consider a late-2020s real-time gross settlement or instant-payments operator migrating under BSI/ANSSI/NCSC guidance. It provisions an SLH-DSA-capable HSM (e.g., Entrust nShield 13.8.0 [26]) whose signing capacity is shared between high-rate ECDSA-P256 settlement traffic and a lower-rate SLH-DSA tier retained for conservative root/offline signing. If the per-tier allowlist is absent (A1, Section 4) or a client advertises only SLH-DSA in its `signature_algorithms` (A3), handshake signing is routed onto the SLH-DSA path; at the calibrated two-worker pool and $\lambda = 13.5$ TPS, this drives utilisation from $\rho \approx 1.44 \times 10^{-4}$ to $\rho = 1.787 > 1$ (Section 5.1), saturating settlement signing with no traffic volume beyond a standards-compliant handshake. This is precisely the pre-deployment gap the defences in Section 7 close before such a pool goes live.

8.2. Becoming in Scope

- **FIPS 206 (FN-DSA) finalisation.** Expected late 2026 or early 2027. FN-DSA is cheaper per-operation than SLH-DSA but still more expensive than ECDSA (see measured values below). Our measurements cover ML-DSA-65, the closest standardised lattice-signature comparator available in liboqs 0.14.0: $\text{CAF}_{r_{\text{cpu}}}(\text{ML-DSA-65, ECDSA-P256}) \approx 2.8\times$ (median sign times $60.4\ \mu\text{s}$ vs. $21.4\ \mu\text{s}$, $n = 50$; Apple M-series ARM64, liboqs 0.14.0; see Table 1). Our benchmarks additionally show FN-DSA-512 at $5.3\times$ and FN-DSA-1024 at $10.7\times$ (Falcon-512/1024 via liboqs 0.14.0). FN-DSA finalisation widens the exposed

surface: deployments that exclude SLH-DSA on DoS grounds but permit FN-DSA will be exposed to $\text{CAF}_{\text{cpu}} \approx 5\text{--}14\times$ (Table A1), insufficient for queue saturation at typical arrival rates but non-trivial for capacity-constrained HSM configurations.

- **Production PQ-signature TLS:** once CDNs issue PQ-signed certificates (signalled for 2026 [1]), the mainline web becomes signature-side exposed (Merkle-tree cert-chain batching is a mitigating IETF direction).
- **Composite schemes:** composite ML-DSA with RSA/ECDSA [28], once at scale, opens a composite-to-SLH-DSA fallback that expands A3's reach.
- **Broader HSM coverage:** as payment-HSM vendors follow Entrust (Futurex has signalled PQ [27]; Thales ships ML-DSA/ML-KEM without SLH-DSA), the A1 surface widens.

8.3. Out of Scope Today

- **Mainline web TLS** (Cloudflare, Chrome, most CDN-terminated domains): ML-KEM-only, “not a single public post-quantum certificate” [1], so the signature-side CAF does not apply today.
- **Chrome** (KEM only: X25519MLKEM768, no PQ signatures [20]) and **Apple iMessage PQ3** (ML-KEM + ECDH ratchet [52]).
- **AWS KMS/ACM/Secrets Manager.** Hybrid KEM is GA [21] and ML-DSA signing is GA [22], but the latter is a rate-limited signing API, not TLS-handshake selection.
- **CNSA 2.0 NSS for general use:** ML-KEM and ML-DSA only, SLH-DSA excluded [49]—a narrower surface than EU/UK.

Adoption trajectory

These signals compose into a consistent 2026–2032 window: KEM-side PQ is already majority traffic [1], AWS has shipped ML-DSA signing [22,23], Cloudflare has signalled PQ-signed certificates for 2026 [1], and regulator and financial-sector roadmaps [46,48,50,51] place high-assurance migration in this window, with FIPS 206 adding a second in-scope family. The exposure window opens at the first production SLH-DSA (or FN-DSA) certificate—this decade, not a speculative future—the pre-deployment moment at which the defences in Section 7 are cheapest to mandate.

8.4. Threat-Realism Summary

Of the four realism conditions in Section 3.5, three are deployment properties that operators control: algorithm in supported set, adversary reach, and utilisation headroom. The fourth (per-operation cost dominating per-trigger attacker cost) is an algorithmic property determined by NIST's standardisation choices. The last is fixed; the first three move with deployment decisions. Our defences (Section 7) address the first two (constraining the supported set at ingress, bounding selection by algorithm-keyed rate limits); the third is a capacity-planning concern.

8.5. Empirical Scope Confirmation

The non-exploiting TLS census in Section 6.1 found zero post-quantum certificates across 692 successful handshakes, which bounds the *out-of-scope* mainline-web surface from an independent vantage and corroborates Cloudflare's October 2025 figure [1]. That census does not sample the in-scope populations above (code-signing, BSI/ANSSI/NCSC high-assurance, financial-sector infrastructure), which public web-port scanning cannot reach, so we draw no inference about in-scope exposure from it. The threat is *latent*: present in standards, reference implementations, and vendor roadmaps, but not yet in mainline production. The window to deploy the defences in Section 7 *before* migration, rather than retroactively, remains open.

8.6. Why Not Simply Mandate ML-DSA and Exclude SLH-DSA?

Since ML-DSA’s CAF is only 1.4–6.2×, why not mandate it and drop SLH-DSA? We give three reasons. (i) *The client selects, not the server*: an adversary advertising *only* SLH-DSA forces the server to sign with it or abort—and aborting is itself a denial of service; the server cannot distinguish a malicious from a constrained-but-legitimate client. (ii) *Compliance*: BSI/ANSSI/NCSC deployments are expected to support all NIST families, including SLH-DSA (FIPS 205), so refusing it may fail an audit (CNSA 2.0’s narrower ML-DSA-only stance does not bind EU/UK/Australian infrastructure). (iii) *Legitimate niche use*: SLH-DSA’s conservative security makes it preferred for offline root-CA signing, and a deployment that keeps it for that purpose while also exposing a TLS endpoint inadvertently creates the exposure we describe. The defences of Section 7 are designed for exactly this: they let SLH-DSA remain in the supported set while preventing the exploitation of its cost asymmetry.

8.7. Why Now

The paper is deliberately positioned at a moment when PQ-signature deployment decisions are being made but have not yet hardened into long-running certificates, multi-year HSM contracts, and regulator-mandated algorithm sets. The defences we measure are cheapest to introduce at this stage; retrofitting them into deployed long-lived cryptographic infrastructure is more expensive. No production PQ-signature DoS incident has been reported in 2024–2026; we argue this is the correct window for forewarning, not forensics.

9. Related Work

This paper sits at the intersection of three research threads: PQ-TLS performance analysis, cryptographic-protocol denial-of-service attacks, and standards-community treatments of hybrid-PQ deployment. Table 4 systematically positions this paper’s contribution against the most closely related prior attack classes.

Table 4. Systematic comparison of existing cryptographic and network DoS attack classes against the contribution of this paper. *Mechanism* distinguishes intra-algorithm from inter-algorithm amplification. *Privilege* is the minimum adversary capability required. *PQ-specific* indicates whether the attack surface exists only in hybrid post-quantum deployments.

Work	Mechanism	Protocol	Privilege	PQ-Spec.	Amplification
sslsqueeze/THC-SSL-DOS [14,15]	RSA private-key asymmetry	TLS (classical)	Network	No	6–15×
QFAM [35]	Attacker-chosen high-cost cipher in QUIC	QUIC	Network	No	~4.6×
KeyTrap [17]	Crafted records force quadratic key×sig validation	DNSSEC	Network	No	~2 × 10 ⁶ ×
DNS FLaRE [53]	Flush-reload side-channel on DNS forwarder cache	DNS	Local/net	No	Cache poisoning
QPADL [40]	Spectrum-access DoS via location spoofing	SAS	Network	Partial	DoS on SAS DB
This work	Inter-algorithm substitution forces SLH-DSA in hybrid PQ negotiation	TLS 1.3	Network	Yes	12,255×

9.1. PQ-TLS Performance Studies

The PQ-TLS literature has emphasised *operational* performance. Sikeridis et al. [5] benchmarked PQ signature algorithms in TLS 1.3 and proposed mixing them across cert chains; Paquin et al. [6] built a network-emulation framework; Crockett et al. [7] prototyped PQ and hybrid key exchange (the origin of Open Quantum Safe); Kampanakis et al. [8]

characterised PQ X.509 viability; and Schwabe et al. [54] proposed KEMTLS to cut server-side signature cost by $\sim 90\%$ —recognising PQ-signature cost as a systems concern, but at the protocol-design level rather than adversarially. More recently, Chen et al. [55] proved tighter PQ-TLS handshake reductions, Lim et al. [56] accelerated handshakes via DNS pre-fetch, and Siddiqua and Yousuf [57] compared classical SSH signatures—all taking performance as given rather than as an attack surface. We ask instead how an adversary exploits the choices deployment exposes.

9.2. Classical SSL/TLS Computational-Asymmetry DoS

sslsqueeze [14] and THC-SSL-DOS [15] exploit RSA-handshake server asymmetry ($6\text{--}15\times$); Bernat [32] surveyed mitigations. These establish the threat class CAF generalises, bounded by server private-key work versus the client’s trivial ClientHello; we push the magnitude two to three orders higher by bringing SLH-DSA into the negotiation surface.

The classical literature did not consider algorithm-choice asymmetry as the primitive because RSA-to-ECDHE and RSA-to-ECDSA migrations shrank server-side handshake cost to a baseline that left little to forcibly inflate. Hybrid-PQ negotiation re-introduces large per-operation cost asymmetries that algorithmic downgrade can exploit.

9.3. Algorithmic-Complexity DoS in Cryptographic Protocols

Crosby and Wallach [38] introduced the general class of algorithmic-complexity DoS attacks: low-bandwidth adversarial inputs forcing worst-case behaviour. Our metric is a specialisation to cryptographic-handshake algorithm selection.

KeyTrap [17] is the closest peer-reviewed prior work: it reports CPU amplification of $\sim 2 \times 10^6$ on DNSSEC resolvers via crafted records forcing quadratic key $\times$ signature validation. KeyTrap and our work are complementary species of Crosby–Wallach attacks applied to cryptographic protocols. The mechanisms differ: KeyTrap is *intra-algorithm*, forcing disproportionate validation work within a single resolution context via crafted record sets (many signatures and colliding key tags); our attack is *inter-algorithm*, substituting a high-cost signature algorithm for a cheap one in a hybrid negotiation. KeyTrap’s magnitude is larger; our contribution is not magnitude but the demonstration that algorithm substitution in hybrid PQ negotiation is a distinct, operationally exploitable DoS class.

Table 5 extends this comparison beyond amplification magnitude to six operationally relevant dimensions. Three dimensions favour this work over KeyTrap. First, adversary access: KeyTrap requires an attacker who controls or can inject into a DNS zone, placing it in a privileged network position; the A3 adversary class in this paper requires only a standard TLS network connection and a crafted ClientHello extension, the lowest-privilege entry point of any amplification attack we surveyed. Second, fix cost: KeyTrap required coordinated changes to resolver validation logic across BIND, Unbound, and PowerDNS [17]; the defences evaluated here (D1 algorithm—selection filtering, D2 algorithm—keyed rate limiting) are sidecar-deployable without modifying the TLS stack. Third, pre-deployment timing: KeyTrap was disclosed after resolvers were already in production and attackers had a window of exploitation; this paper analyses the PQ-TLS threat surface before SLH-DSA enters widespread production cert chains, when algorithm-policy choices can still be made defensively.

Zhang et al. [58] identified a persistent class of DoS attacks against DNSSEC validators that exploit the reuse of unvalidated resolver caches (USENIX Security 2025), and Daniluk et al. [59] surveyed NSEC3 parameter configurations in the wild at IMC 2024, showing that operational DNSSEC deployments retain configurations vulnerable to offline hash-guessing. These works establish that DNSSEC-layer algorithmic DoS remains an active research area; the mechanism in our paper is structurally analogous (an adversary-

controlled protocol input forces disproportionate cryptographic work on the server) but applied to TLS 1.3 signature-algorithm selection rather than DNS resolution. Wu et al. [60] demonstrated cross-layer DoS against UDP services on Linux via socket-buffer saturation (CCS 2025), establishing a complementary resource-exhaustion model: in both cases, the attacker does not need to overwhelm network bandwidth but instead exhausts a bounded internal resource (socket buffer vs. HSM signing queue).

QFAM [35] introduced the term *CPU amplification factor* for QUIC handshake flooding (reported $\sim 4.6\times$ on aioquic) and explicitly identified attacker-chosen high-cost algorithms as an attack vector. QFAM’s scope is pure-classical QUIC; our paper specialises the metric to hybrid PQ, signature-verification asymmetry, HSM queue-side resources, and the inter-algorithm substitution mechanism. We adopt the CAF name following QFAM, with explicit lineage acknowledgement (Section 3.6).

Table 5. Multi-dimensional comparison of KeyTrap [17] and this work beyond CPU amplification magnitude. Dimensions are chosen to characterise operational exploitability and defensive tractability independently of the raw amplification factor.

Dimension	KeyTrap (DNSSEC)	This Work (PQ-TLS)
Mechanism	Intra-algorithm: crafted records force quadratic key $\times$ signature validation within one algorithm family	Inter-algorithm: adversary substitutes a high-cost algorithm (SLH-DSA) for a cheap one (ECDSA) in hybrid negotiation
Adversary access required	DNS zone control or record injection (privileged network position)	Standard TLS network connection; crafted ClientHello only (A3: no credentials, no insider access)
CPU amplification	$\sim 2,000,000\times$	$12,255\times$ (ARM64); $14,272\times$ (Graviton3)
Deployment surface today	All DNSSEC-validating resolvers (universally deployed)	Hybrid PQ deployments with SLH-DSA in cert chain (narrow today; widens as FIPS 206 finalises)
Detection at wire level	Query traffic indistinguishable from legitimate DNSSEC resolution	ClientHello indistinguishable from a legitimate client advertising SLH-DSA preference
Patch status	CVE-2023-50387/-50868; patched in all major resolvers (BIND, Unbound, PowerDNS)	No protocol CVE (exploits intended TLS 1.3 behaviour); defences (D1/D2/PSA) available but not yet standardised
Fix deployment cost	Resolver validation logic changes; coordinated multi-vendor patch cycle	Algorithm-selection filter or rate limiter; sidecar-deployable without modifying TLS stack
Pre-deployment warning	No: disclosed after active deployment and exploitation window	Yes: this paper analyses the threat before SLH-DSA enters widespread production cert chains

9.4. Downgrade-Attack Taxonomy

Bhargavan et al. [42] formalised a taxonomy of downgrade attacks in key-exchange protocols (element, target, vector, damage) and proved TLS 1.3’s resilience against classical downgrade. Lee et al. [43] demonstrated that middlebox-induced fallback still produces TLS 1.3 version- downgrade surfaces. The Bhargavan taxonomy treats *damage* as *weakening* (confidentiality or authentication loss). Our attack inflicts a different kind of damage: *cost inflation*. Unlike a downgrade, the adversary forces selection of a *more* expensive (and, here, higher-security-level) algorithm, and the negotiation’s integrity remains intact (Section 4.5); the abuse is of selection, not of transcript integrity. The taxonomy’s weakening-centred damage axis does not, to our knowledge, have a cell for this “downgrade-to-expensive” variant, so we position it as an extension of the framework rather than a duplication.

9.5. Hybrid PQC Strategy Surveys

The SoK of Fall [61] systematised hybrid PQ deployment strategies for TLS 1.3, QUIC, and SSH, surveying 32 works on confidentiality guarantees and standards activity. It touches downgrade risk within the confidentiality frame. Our paper is the orthogonal *availability* treatment: we look at what hybrid-PQ deployment exposes when the attacker's goal is DoS rather than key recovery.

Benčina et al. [62] provided a formal cryptographic analysis of hybrid-PQ SSH; Lee et al. [63] performed a system-level study of quantum-resistant DNS, noting DoS amplification qualitatively in one table row without formalisation. Rawat [64] proposed DNSSEC without signatures, motivated in part by PQ signature-size and DoS concerns. The Crosby–Wallach framing is active in this adjacent literature but has not been applied to PQ signature substitution in TLS.

9.6. Standards-Community Treatment of PQ DoS

Several IETF drafts explicitly address PQ-DoS subtypes, all centred on bandwidth amplification rather than compute amplification. The certificate-compression draft [12] compresses WebPKI certificates to mitigate PQ-certificate bandwidth impact; the key-share-prediction draft [19] motivates itself on PQ KEM “excessive bandwidth”; the UTA application draft [13] names a 5–8× bandwidth amplification as a DoS risk. The hybrid-signature spectrum draft [65] and the dual-cert draft [44] discuss the negotiation surface our attack exploits, without treating availability adversarially.

Among regulator advisories surveyed (NSA CNSA 2.0, CISA, NCSC-UK, BSI, ANSSI, ENISA, NCSC-NL), only the CISA Oct 2024 document *Post-Quantum Considerations for OT* [66] explicitly names PQ-induced DoS as a concern; even that document does not address algorithm substitution. We emphasise this *standards gap*: to the best of our knowledge, the CPU-cost-asymmetry adversarial-selection threat we describe is not treated as a first-class concern in any active IETF TLS/PQUIP/LAMPS working-group draft we surveyed (October 2025). This gap claim is bounded by an explicit survey scope: the active TLS, PQUIP, and LAMPS working-group drafts together with the regulator advisories enumerated above, read specifically for adversarial algorithm selection and compute-amplification DoS as of October 2025. We assert absence within that surveyed corpus, not universal non-existence.

9.7. Client-Puzzle Defences

Client puzzles [36,37] impose *defender*-side asymmetry; CAF measures the mirror image (attacker-imposed). Our D1 re-introduces the puzzle framing along the algorithm-choice axis: the high-CAF algorithm becomes a restricted tier, with legitimate use routed to a low-capacity lane.

9.8. Implementation-Side Availability Bugs

CVE-2024-37305 [31] (an oqs-provider hybrid-key deserialisation overflow) and CVE-2024-30172 [67] (a Bouncy Castle Ed25519 verification infinite loop, a classical Crosby–Wallach DoS) show PQ-implementation availability bugs are real, but the class we study is *design-level* (intended TLS 1.3 behaviour), not implementation-level.

9.9. Positioning

Our contribution sits one layer up from Crosby–Wallach, across from KeyTrap, one specialisation beyond QFAM, and orthogonal to the PQ-TLS performance literature. The contribution is the combination: a formal metric, working PoCs across two deployments, measured defences, and a standards-gap argument that, to the best of our knowledge,

current IETF drafts have not yet addressed. We state the basis of the novelty claim precisely, to avoid overstatement. It is *not* the amplification magnitude—KeyTrap’s is roughly two orders larger—and *not* the algorithmic-complexity genus of Crosby–Wallach [38], of which this attack is a specialisation. The novelty is their conjunction: (i) the amplification primitive is *inter-algorithm* substitution rather than intra-algorithm work inflation; (ii) it is bound to an explicit queue-stability predicate, $\rho = \lambda / (c\mu) > 1$, rather than reported as a per-operation ratio; and (iii) it is reachable from the lowest-privilege position of any amplification attack we surveyed—a standard TLS ClientHello, with no zone control, credential, or insider access. No single one of these is claimed as novel in isolation.

10. Limitations and Future Work

We list limitations explicitly; several pre-empt anticipated objections and several are genuine open questions.

Single-jurisdiction deployment context

The NPP-representative testbed is calibrated on Australian payment volumes. The financial-systems threat surface varies by jurisdiction; our headline numbers generalise to equivalent two-worker HSM configurations but not to larger settlement infrastructures (TARGET2, CHAPS) without recalibration. We report the canonical CAF table per-microarchitecture to enable that recalibration.

HSM realism

Our PoC runs against SoftHSMv2 2.7.0 (a software HSM), not hardware HSMs (the canonical table, Table A1, spans five microarchitectures but no dedicated silicon). Hardware acceleration shifts the absolute ratio but not the qualitative threat: SLH-DSA’s cost is dominated by sequential hypertree hashing, which an FPGA/ASIC hash engine [18,26] can cut by one to two orders—but the same silicon does not speed ECDSA proportionally, so $\text{CAF}_{r_{\text{cpu}}}$ falls from $\sim 10^4$ to perhaps $\sim 10^2$ – 10^3 , not to single digits, and the stability inequality $\lambda > c\mu$ is unchanged (offload only moves the arrival-rate threshold; thermal throttling under load raises the ratio). A real-hardware cross-check is underway (Thales Luna for ML-DSA, Entrust nShield 13.8.0 for SLH-DSA); a full multi-vendor matrix is future work.

We therefore present these results as proof-of-concept evidence of the mechanism and its magnitude on OQS/OpenSSL and SoftHSMv2 testbeds, not as exact production measurements. A commercial HSM may exhibit different absolute latencies, and a production deployment may interpose additional bottlenecks (network, API, admission control) or mitigations (vendor rate limiting, algorithm-tier policies) that this testbed does not model. What the measurements establish is the qualitative threat: once SLH-DSA enters a shared signing path, the queue-instability condition $\lambda > c\mu$ becomes reachable at realistic arrival rates, and the per-operation cost ratio that drives it is structural rather than implementation-specific.

Testbed scale versus production

Beyond HSM silicon, the testbed differs from a large deployment in structure: it is a single host with $c = 2$ workers under a controlled, scripted load, whereas production payment infrastructure adds load balancers, autoscaling signing pools, multi-tenant contention, and genuinely diurnal, bursty arrivals. These factors do not cancel the threat: a larger worker count c and autoscaling raise the arrival-rate threshold $c\mu$ but leave the per-handshake cost asymmetry and the $\rho > 1$ reachability intact—an all-SLH-DSA client still forces the expensive path, and it is the admission bound of Proposition 1, not pool size, that caps ρ . Multi-tenant contention lowers the effective per-lane μ and so raises ρ , while burstier-than-Poisson arrivals worsen queueing delay at any given ρ (Section 2.6); both push the undefended system further into saturation. The production gap therefore rescales the threshold, not the mechanism. We

separate the conclusions that survive this gap from those that do not. *Robust*: the per-operation $\text{CAF}_{r_{\text{cpu}}}$ is a hardware-measured primitive (Table A1); the $\rho > 1$ saturation it induces is distribution-free—independent of arrival burstiness and service variability (Section 2.6)—and scale-free under the admission bound of Proposition 1, which extends to clustered HSMs (Section 7.4). *Testbed-specific*: the absolute queue-drain latencies, the D3 detection and circuit-break timings (which never fired because the host out-paced the benchmark μ), and the exact overhead percentages—all of which need hardware-rate-matched production validation. The attack’s existence and order of magnitude are thus established here; their precise operational tuning is deployment-dependent.

M/M/c queueing assumptions

The analysis uses M/M/c. Section 2.6 quantifies the two deviations from it: SLH-DSA service is near-deterministic ($C_s^2 \approx 3.8 \times 10^{-4}$, $\text{SD}/\text{mean} = 1.9\%$, $N = 8$; Section 5.1) and payment traffic is burstier than Poisson. The two oppose—by Pollaczek–Khinchine the deterministic-service correction $(1 + C_s^2)/2 \approx 0.5$ makes M/M/c conservative by $\approx 2\times$ below saturation, while the Allen–Cunneen factor $(C_a^2 + C_s^2)/2$ shows heavy burstiness, which pushes the queue the other way. For the operative $\rho > 1$ regime, the queue grows unbounded for *any* arrival/service distribution (Loynes’ criterion), so the saturation claim is distribution-free; we use M/M/c for its closed-form bounds.

liboqs version pinning

We pin liboqs 0.14.0 and oqs-provider 0.11.0/OpenSSL 3.6.0 for reproducibility; later versions will optimise SLH-DSA further [18]. CAF is version-relative: a $2\text{--}3\times$ library speedup lowers it correspondingly, but SLH-DSA’s hypertree structure floors the per-operation cost, so the threat class persists.

Single-attack-origin PoCs

Our measurements run the A3 attack from one origin (and briefly from two, to verify cloud-region independence). Real distributed adversaries would amplify the visible impact further. We do not model distributed adversaries; the amplification would only grow, so our numbers are conservative in that sense. This will be future work.

Defence limits

Three constraints bound the defences, detailed in Sections 7.7 and 7.3: D3’s detection latency is environment-dependent (on our compute-rich testbed the circuit-break never fired, as the host stayed below $\rho > 1$); D3’s fixed thresholds are not robust to a low-and-slow attacker, against which the hard admission controls D1 and D2 remain effective; and D1 requires ClientHello visibility that Encrypted ClientHello [45] removes unless TLS terminates at a controlled edge—the configuration typical of our in-scope high-assurance deployments.

KEM-side CAF is treated only in passing

KEM-side asymmetry across NIST-standardised algorithms is single-digit and does not produce order-of-magnitude amplification. The canonical CAF table (Table A1) covers signature algorithms; ML-KEM timing data is included in the extended reproducibility artefact. The paper’s narrative stays on the signature side.

Internet scan scope

Our internet-scale passive TLS census (Section 6.1) covers Tranco top-1000 and a curated 35-target high-value list. It is not a stratified population sample; larger-scale automated census with RIPE Atlas or similar infrastructure is future work. The *absence* of

PQ signatures in 692 successful connections bounds prevalence only on the *out-of-scope* high-traffic public-web surface (consistent with Section 6.1 and the Cloudflare figure); by construction, it is not evidence about the in-scope populations (code-signing, high-assurance, financial), which public web scanning cannot reach. We draw no in-scope inference from it.

Further scope limits

We model rather than build the A2 supply-chain exploit (the compromise mechanism is out of scope). Composite signatures [28] are a natural CAF extension—combined cost is additive, so the class carries with shifted magnitudes—but are not treated here. Our single-core steady-state measurements omit multi-core cache contention and thermal throttling, both of which would *amplify* CAF. And a saturated HSM queue triggers higher-layer retries (application, interbank settlement) whose systemic cascade is left to follow-up work.

Open questions for the community include the following:

- Should IETF drafts for PQ-signature-using protocols incorporate per-tier algorithm-capacity policies as normative text?
- Should NIST publish per-algorithm throughput-vs-DoS-resilience guidance as part of its PQ migration materials?
- Should HSM vendors ship default policy templates that bind SLH-DSA to an isolated capacity pool?
- Is there an algorithmic alternative to SLH-DSA that preserves its conservative-security property (reduction to second-preimage and target-collision resistance rather than full collision resistance) at a lower per-operation cost? Cert-chain batching (an active IETF TLS WG direction) reduces signature-verification load at the TLS layer but does not address per-sign cost at the server.

11. Conclusions

Hybrid post-quantum TLS deployment has been studied as an operational problem—latency, bandwidth, certificate size—leaving its availability surface unexamined. An adversary with low privilege (a normal TLS client, a misconfigured deployment script, or a compromised library-distribution pipeline) can exploit the intended algorithm-selection behaviour of TLS 1.3 to force a victim onto SLH-DSA, a CPU amplification of $3904\text{--}14,272\times$ across five microarchitectures (Table A1): two to three orders of magnitude above the classical handshake-asymmetry baseline of $6\text{--}15\times$. The threat class is distinct from intra-family validation explosion (KeyTrap) and pure-classical handshake amplification (QFAM); to the best of our knowledge, this is the first systematic quantification of CPU-cost adversarial algorithm substitution in hybrid post-quantum negotiation, with queue-bounding defences.

The Cryptographic Amplification Factor (CAF) makes the asymmetry measurable. Four realism conditions bound when $\text{CAF}_{r_{\text{cpu}}} \gg 1$ maps to an operational denial-of-service. Working PoCs on an NPP-representative testbed (A1) and on OQS-OpenSSL 3 with oqs-provider (A3) show that the attack transitions from arithmetic to operational under realistic configurations. The saturation itself is distribution-free—it holds under M/M/c, M/D/c, and bursty-arrival refinements alike (Section 2.6), so the result does not rest on the queueing model's simplifying assumptions. Two enforcement defences (D1 selection enforcement, D2 algorithm-keyed rate limiting) plus the composite Parallel Signing Architecture bound $\rho_{\text{SLH-DSA}} \leq \alpha < 1$ by admission control (established formally in Proposition 1), preventing queue saturation regardless of attack arrival rate. D3 (CAF-spike anomaly detection) is an early-warning layer rather than a hard bound: its detection latency is environment-

dependent (degraded when the deployment's actual service rate exceeds the calibration μ , as observed on our compute-rich testbed where the circuit-break threshold did not fire). Against a low-and-slow attacker who paces requests to evade D3's thresholds, the hard admission controls D1 and D2 remain effective, so the defence-in-depth guarantee does not rest on D3. Deployment cost is modest: D1/D2/D3 are sidecar-scale; PSA adds one isolated low-throughput signing tier (a single additional certified HSM slot or worker pool, which we do not quantify in dollar terms).

The threat surface today is bounded: mainline web TLS is KEM-only and unexposed. The surface widens as BSI/ANSSI/NCSC-aligned high-assurance deployments, software/firmware signing systems, and financial-sector PQ migration programmes bring SLH-DSA into production cert chains and HSM configurations. FIPS 206 (FN-DSA) finalisation in late 2026 or early 2027 widens it further.

The IETF working groups have addressed PQ bandwidth-side DoS (certificate compression, key-share prediction, the UTA application draft) but, to the best of our knowledge, have not addressed CPU-cost adversarial algorithm substitution as a first-class threat. Among regulators surveyed, only CISA's October 2024 operational-technology guidance names PQ-induced DoS at all. This paper articulates that gap and provides the metric, PoCs, and defence vocabulary needed to close it.

Two lessons generalise beyond the specific attack. First, hybrid cryptographic negotiation surfaces that mix algorithms with very different per-operation costs create availability threats that existing downgrade-attack taxonomies do not capture: downgrade-to-expensive is a distinct damage dimension. Second, deployment decisions made in the current PQ transition window are expensive to change once baked into long-running certificates, multi-year HSM contracts, and regulator-mandated algorithm sets; analysis at the moment of decision is materially cheaper than retrofitted analysis later. We offer this paper in that spirit.

Author Contributions: N.S.S.: Conceptualisation, methodology, software, validation, formal analysis, investigation, data curation, visualisation, writing the original draft preparation. S.A.L.: Methodology, validation, investigation, revising based on review feedback and editing. R.N.: Reviewing and editing, supervision, project administration, interpretation of results, and critical revision of the manuscript. S.B.: Review and editing, supervision, project administration, scholarly guidance, and providing critical comments for the revision of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding. Cloud benchmark measurements (EC2 c6i/c5/c6a/c7g, ap-southeast-2, 18 April 2026) were funded by the authors; no academic cloud research credits were used.

Data Availability Statement: The data and code supporting this study—measurement pipelines, defence implementations, and data-reduction scripts—are available from the corresponding author on reasonable request. The attack proof-of-concept code operates only against author-controlled testbed infrastructure and is shared with verified researchers on request, in line with responsible-disclosure practice.

Acknowledgments: The Open Quantum Safe project maintainers are thanked for the liboqs and oqs-provider open-source releases that made this work possible.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Canonical CAF Table (Multi-Architecture)

Table A1 reports $\text{CAF}_{r_{\text{cpu}}}$ across five microarchitectures (baseline ECDSA-P256, median signing latency; `expanded_bench.py`, 18 April 2026). Apple M-series (ARM64, Apple M3 Max): $n = 50$, 5 warm-up iterations discarded, liboqs 0.14.0, macOS. Cloud x86-64/Graviton3: $n = 500$, 3 warm-up iterations discarded, liboqs 0.14.0, Ubuntu 22.04, EC2 ap-southeast-2, `cpupower` performance governor, Intel turbo disabled.

Table A1. Canonical $\text{CAF}_{r_{\text{cpu}}}$ across five microarchitectures (baseline: ECDSA-P256, median signing latency; `expanded_bench.py`, 18 April 2026). Apple M-series (ARM64, Apple M3 Max): $n = 50$, 5 warmup iterations discarded, liboqs 0.14.0, macOS. Cloud x86-64/Graviton3: $n = 500$, 3 warmup iterations discarded, liboqs 0.14.0, Ubuntu 22.04, EC2 ap-southeast-2; cpupower performance governor, Intel turbo disabled.

Algorithm	NIST Std.	Apple M-Series (ARM64)	Ice Lake (x86-64)	Cascade Lake (x86-64)	EPYC Milan (x86-64)	Graviton3 (ARM64)
ECDSA-P256	— (baseline)	1×	1×	1×	1×	1×
ML-DSA-44	FIPS 204	1.8×	1.4×	1.5×	1.7×	3.0×
ML-DSA-65	FIPS 204	2.8×	1.8×	2.4×	2.8×	4.8×
ML-DSA-87	FIPS 204	3.0×	2.5×	2.8×	3.5×	6.2×
FN-DSA-512	FIPS 206 draft	5.3×	6.9×	7.3×	6.0×	5.4×
FN-DSA-1024	FIPS 206 draft	10.7×	13.8×	14.3×	12.0×	10.6×
SLH-DSA-SHA2-128s	FIPS 205	12,255×	4389×	4165×	3904×	14,272×
SLH-DSA-SHA2-128f	FIPS 205	592.2×	214.6×	204.9×	192.0×	688.4×
SLH-DSA-SHA2-192s	FIPS 205	21,746×	8449×	8188×	7368×	25,865×
SLH-DSA-SHA2-256s	FIPS 205	19,344×	7745×	7485×	6756×	22,951×
SLH-DSA-SHAKE-128s	FIPS 205	19,842×	9002×	8211×	8997×	24,944×
SLH-DSA-SHAKE-128f	FIPS 205	979.5×	423.6×	394.8×	415.5×	1201×

Appendix B. D3 Detection Latency: Analytical Bounds and Measurements

Derivation of analytical bounds. An EMA with time constant $\tau = 5$ s rising from $\rho \approx 0$ reaches threshold θ after $t^* = \tau \ln(\rho_{\text{true}}/(\rho_{\text{true}} - \theta))$ seconds, and the “sustained for T s” requirement adds T ; substituting $\rho_{\text{true}} = 1.787$ and each trigger’s (θ, T) gives the per-trigger bounds in the “Analytic” column of Table A2. The same table reports the analytic false-positive bound under benign Poisson SLH-DSA arrivals at $\lambda_{\text{SLH-DSA}} \leq 0.5$ TPS (<0.01 events/day at $z > 4$).

Table A2. D3 detection latency under A1 attack parameters ($\rho_{\text{true}} = 1.787$ from Section 5.1; EMA $\tau = 5$ s, poll 0.5 s). “Analytic” bounds are derived from the EMA step-response at $\rho_{\text{true}} = 1.787$. “Measured” medians are from $N = 5$ live trials on the local testbed (actual $\rho \approx 0.6$; host faster than the assumed μ). “FP rate” is the analytic bound under benign Poisson arrivals. [†] Circuit-break did not fire in any of the 5 trials (host throughput kept $\rho < 0.8$).

Trigger	Threshold	Analytic (s)	Measured Median (s)	FP Rate
ρ -alert	>0.5 for 5 s	≤ 7	18.5 (4/5 trials)	<0.01 /day
ρ -break	>0.8 for 2 s	≤ 5	— [†]	<0.01 /day
z-score alert	$z > 4$	≤ 3	21.3 (3/5 trials)	<0.01 /day

References

1. Cloudflare Research. State of the Post-Quantum Internet in 2025. Cloudflare Blog, 2025. Available online: <https://blog.cloudflare.com/pq-2025/> (accessed on 14 April 2026).
2. FIPS 203; Module-Lattice-Based Key-Encapsulation Mechanism Standard. National Institute of Standards and Technology: Gaithersburg, MD, USA, 2024.
3. FIPS 204; Module-Lattice-Based Digital Signature Standard. National Institute of Standards and Technology: Gaithersburg, MD, USA, 2024.
4. FIPS 205; Stateless Hash-Based Digital Signature Standard. National Institute of Standards and Technology: Gaithersburg, MD, USA, 2024.
5. Sikeridis, D.; Kampanakis, P.; Devetsikiotis, M. Post-Quantum Authentication in TLS 1.3: A Performance Study. In Proceedings of the 27th Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 23–26 February 2020; ePrint 2020/071.
6. Paquin, C.; Stebila, D.; Tamvada, G. Benchmarking Post-Quantum Cryptography in TLS. In *Proceedings of the Post-Quantum Cryptography (PQCrypto)*; LNCS; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12100, pp. 72–91. https://doi.org/10.1007/978-3-030-44223-1_5.

7. Crockett, E.; Paquin, C.; Stebila, D. Prototyping Post-Quantum and Hybrid Key Exchange and Authentication in TLS and SSH. In *Proceedings of the NIST 2nd Post-Quantum Cryptography Standardization Conference*; Cryptology ePrint Archive, Paper 2019/858; NIST Computer Security Resource Center: Gaithersburg, MD, USA, 2019.
8. Kampanakis, P.; Panburana, P.; Daw, E.; Geest, D.V. The Viability of Post-Quantum X.509 Certificates. ePrint 2018/063. 2018. Available online: <https://eprint.iacr.org/2018/063> (accessed on 16 April 2026).
9. Stebila, D.; Fluhrer, S.; Gueron, S. Hybrid Key Exchange in TLS 1.3. Internet-Draft, draft-ietf-tls-hybrid-design-16. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/16/> (accessed on 15 May 2026).
10. IETF TLS WG. ML-KEM Post-Quantum Key Agreement for TLS 1.3. Internet-Draft, draft-ietf-tls-mlkem-07. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-tls-mlkem/07/> (accessed on 15 May 2026).
11. IETF TLS WG. Use of ML-DSA in TLS 1.3. Internet-Draft, draft-ietf-tls-mldsa-03. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-tls-mldsa/03/> (accessed on 15 May 2026).
12. IETF TLS WG. Abridged Compression for WebPKI Certificates. Internet-Draft, draft-ietf-tls-cert-abridge-02. 2024. Available online: <https://datatracker.ietf.org/doc/draft-ietf-tls-cert-abridge/02/> (accessed on 15 May 2026).
13. IETF UTA WG. Post-Quantum Cryptography Recommendations for TLS-Based Applications. Internet-Draft, draft-ietf-uta-pqc-app-01. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-uta-pqc-app/01/> (accessed on 15 May 2026).
14. Trojnara, M. sslsqueeze: SSL Computational Denial of Service Tool. GitHub Repository. Available online: <https://github.com/mmgaggle/sslsqueeze> (accessed on 17 April 2026).
15. The Hacker's Choice. THC-SSL-DOS. Tool Release, 2011. Available online: <https://gitlab.com/kalilinux/packages/thc-ssl-dos> (accessed on 3 May 2026).
16. Sammo, N.S. Post-Quantum Cryptography Migration in Australian Real-Time Payment Infrastructure: A Monte Carlo Simulation Study of the New Payments Platform. *arXiv* **2026**, arXiv:2605.02276. <https://doi.org/10.48550/arXiv.2605.02276>.
17. Heftrig, E.; Schulmann, H.; Vogel, N.; Waidner, M. The Harder You Try, The Harder You Fail: The KeyTrap Denial-of-Service Algorithmic Complexity Attacks on DNSSEC. In *Proceedings of the 31st ACM Conference on Computer and Communications Security (CCS)*, Salt Lake City, UT, USA, 14–18 October 2024. <https://doi.org/10.1145/3658644.3670389>.
18. Saarinen, M.J.O. Accelerating SLH-DSA by Two Orders of Magnitude with a Single Hash Unit. In *Proceedings of the Fifth NIST PQC Standardization Conference*, NIST, Washington, DC, USA, 10–12 April 2024.
19. IETF TLS WG. TLS Key Share Prediction. Internet-Draft, draft-ietf-tls-key-share-prediction-04. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-tls-key-share-prediction/04/> (accessed on 15 May 2026).
20. Chromium Project. TLS 1.3 Hybridized Kyber Support (X25519Kyber768/X25519MLKEM768). Chrome Platform Status & Enterprise Release Notes, 2024. Available online: <https://blog.chromium.org/2023/08/protecting-chrome-traffic-with-hybrid.html> (accessed on 8 April 2026).
21. AWS Security Blog. ML-KEM Post-Quantum TLS Now Supported in AWS KMS, ACM, and Secrets Manager, 2025. Available online: <https://aws.amazon.com/blogs/security/ml-kem-post-quantum-tls-now-supported-in-aws-kms-acm-and-secrets-manager/> (accessed on 21 April 2026).
22. AWS Security Blog. How to Create Post-Quantum Signatures Using AWS KMS and ML-DSA, 2025. Available online: <https://aws.amazon.com/blogs/security/how-to-create-post-quantum-signatures-using-aws-kms-and-ml-dsa/> (accessed on 24 April 2026).
23. AWS Security Blog. Post-Quantum (ML-DSA) Code Signing with AWS Private CA and AWS KMS, 2025. Available online: <https://aws.amazon.com/blogs/security/post-quantum-ml-dsa-code-signing-with-aws-private-ca-and-aws-kms/> (accessed on 30 April 2026).
24. Microsoft Security Blog. Microsoft's Quantum-Resistant Cryptography Is Here, 2024. Available online: <https://techcommunity.microsoft.com/blog/microsoft-security-blog/microsofts-quantum-resistant-cryptography-is-here/4238780> (accessed on 5 May 2026).
25. Thales. Luna HSM Firmware v7.9: Native ML-KEM and ML-DSA Support. Thales Product Release, 2025. Available online: <https://cpl.thalesgroup.com/blog/encryption/luna-hsm-pqc-quantum-safe-encryption> (accessed on 12 May 2026).
26. Entrust. nShield 5 HSM: FIPS 140-3 L3 Validation and ML-DSA/ML-KEM Firmware, 2025. Available online: <https://www.entrust.com/company/newsroom/entrust-nshield-hsms-achieve-validation-from-nist-cryptographic-algorithm-validation-program> (accessed on 15 April 2026).
27. Futurex. Payment HSM Encryption Trends for 2026 and Beyond. Futurex Blog, 2025. Available online: <https://www.futurex.com/blog/payment-hsms-trends> (accessed on 19 May 2026).
28. IETF LAMPS WG. Composite ML-DSA for Use in Internet PKI. Internet-Draft, draft-ietf-lamps-pq-composite-sigs-19. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-lamps-pq-composite-sigs/19/> (accessed on 15 May 2026).
29. IETF LAMPS WG. SLH-DSA Algorithm Identifiers for X.509. Internet-Draft, draft-ietf-lamps-x509-slhdsa-09. 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-lamps-x509-slhdsa/09/> (accessed on 15 May 2026).

30. IETF LAMPS WG. Internet X.509 Public Key Infrastructure—Algorithm Identifiers for ML-DSA. Internet-Draft, draft-ietf-lamps-dilithium-certificates-13. 2024. Available online: <https://datatracker.ietf.org/doc/draft-ietf-lamps-dilithium-certificates/13/> (accessed on 15 May 2026).
31. NVD. CVE-2024-37305: Oqs-Provider Hybrid Key Length Handling Vulnerability. National Vulnerability Database, 2024. Patched in oqs-Provider v0.6.1. Available online: <https://nvd.nist.gov/vuln/detail/CVE-2024-37305> (accessed on 2 May 2026).
32. Bernat, V. TLS Computational DoS Mitigation. Blog Post, 2011. Available online: <https://vincent.bernat.ch/en/blog/2011-ssl-dos-mitigation> (accessed on 10 May 2026).
33. Loynes, R.M. The stability of a queue with non-independent inter-arrival and service times. *Math. Proc. Camb. Philos. Soc.* **1962**, *58*, 497–520. <https://doi.org/10.1017/S0305004100036781>.
34. Allen, A.O. *Probability, Statistics, and Queueing Theory with Computer Science Applications*, 2nd ed.; Academic Press: San Diego, CA, USA, 1990.
35. Jabbari, A.; Joarder, Y.A.; Teyssier, B.; Fung, C. QFAM: Mitigating QUIC Handshake Flooding Attacks Through Crypto Challenges. *arXiv* **2024**, arXiv:2412.08936. <https://doi.org/10.48550/arXiv.2412.08936>.
36. Juels, A.; Brainard, J. Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*; Internet Society: Fredericksburg, VA, USA, 1999; pp. 151–165.
37. Stebila, D.; Kuppusamy, L.; Rangasamy, J.; Boyd, C.; Nieto, J.M.G. *Stronger Difficulty Notions for Client Puzzles and Denial-of-Service-Resistant Protocols*; Springer: Berlin/Heidelberg, Germany, 2010.
38. Crosby, S.A.; Wallach, D.S. Denial of Service via Algorithmic Complexity Attacks. In *Proceedings of the 12th USENIX Security Symposium*, Washington, DC, USA, 4–8 August 2003.
39. Li, Q.; Ma, B.; Wang, X.; Wang, C.; Gao, S. Image Steganography in Color Conversion. *IEEE Trans. Circuits Syst. II Express Briefs* **2024**, *71*, 106–110. <https://doi.org/10.1109/TCSII.2023.3300330>.
40. Darzi, S.; Nouma, S.E.; Sedghighadikolaei, K.; Yavuz, A.A. QPADL: Post-Quantum Private Spectrum Access with Verified Location and DoS Resilience. *arXiv* **2025**, arXiv:2510.03631.
41. Australian Prudential Regulation Authority. *CPS 234: Information Security Standard*; Technical Report; Australian Prudential Regulation Authority: Sydney, Australia, 2019.
42. Bhargavan, K.; Brzuska, C.; Fournet, C.; Green, M.; Kohlweiss, M.; Zanella-Béguelin, S. Downgrade Resilience in Key-Exchange Protocols. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*; IEEE: Piscataway, NJ, USA, 2016; pp. 506–525. <https://doi.org/10.1109/SP.2016.37>.
43. Lee, S.; Shin, Y.; Hur, J. Return of Version Downgrade Attack in the Era of TLS 1.3. In *Proceedings of the 16th ACM CoNEXT*, Barcelona, Spain, 1–4 December 2020. <https://doi.org/10.1145/3386367.3431310>.
44. IETF TLS Individual. PQ/T Hybrid Authentication with Dual Certificates. Internet-Draft, draft-yusef-tls-pqt-dual-certs-01. 2025. Available online: <https://datatracker.ietf.org/doc/draft-yusef-tls-pqt-dual-certs/01/> (accessed on 15 May 2026).
45. Rescorla, E.; Oku, K.; Sullivan, N.; Wood, C.A. TLS Encrypted Client Hello. RFC 9849, 2026. Available online: <https://doi.org/10.17487/RFC9849> (accessed on 20 April 2026).
46. Bundesamt für Sicherheit in der Informationstechnik. *Cryptographic Mechanisms: Recommendations and Key Lengths (TR-02102-1, v2025-01)*; Technical Report; Bundesamt für Sicherheit in der Informationstechnik: Bonn, Germany, 2025.
47. Agence nationale de la sécurité des systèmes d’information. *Follow-Up Position Paper on Post-Quantum Cryptography*; Technical Report; Agence nationale de la sécurité des systèmes d’information: Paris, France, 2023.
48. UK National Cyber Security Centre. *Timelines for Migration to Post-Quantum Cryptography*; Technical Report; UK National Cyber Security Centre: London, UK, 2025.
49. National Security Agency. *Commercial National Security Algorithm Suite 2.0 (CNSA 2.0)*; Technical Report; National Security Agency: Meade, MD, USA, 2022.
50. Bank for International Settlements. *Quantum-Readiness for the Financial System: A Roadmap*; Technical Report Papers No. 158; Bank for International Settlements: Basel, Switzerland, 2025.
51. G7 Cyber Expert Group. *G7 Cyber Expert Group Statement on Advancing a Coordinated Roadmap for the Transition to Post-Quantum Cryptography in the Financial Sector*; Technical Report; G7 Cyber Expert Group: Sicily, Italy, 2026.
52. Apple Security Engineering and Architecture. iMessage with PQ3: The New State of the Art in Quantum-Secure Messaging at Scale. Apple Security Research Blog, 2024. Available online: <https://security.apple.com/blog/imessage-pq3/> (accessed on 28 April 2026).
53. Moav, G.; Afek, Y.; Bremner-Barr, A.; Klein, A. DNS FLARE: A Flush-Reload Attack on DNS Forwarders. In *Proceedings of the 34th USENIX Security Symposium (USENIX Security 25)*, Seattle, WA, USA, 13–15 August 2025.
54. 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)*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 1461–1480. <https://doi.org/10.1145/3372297.3423350>.

55. Chen, J.; Zhou, B.; Chen, R.; Jiang, H.; Wang, Y.; Huang, X.; Zhao, Y.; Yung, M. Post-Quantum TLS 1.3 Handshake from CPA-Secure KEMs with Tighter Reductions. In *Proceedings of the Advances in Cryptology—EUROCRYPT 2026*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2026; Volume 16542, pp. 499–530. https://doi.org/10.1007/978-3-032-25317-0_17.
56. Lim, S.; Lee, H.; Jeong, G.; Kwon, T.T. PQTLS-AD: Post-Quantum TLS Accelerated with DNS. In *Proceedings of the 2025 34th International Conference on Computer Communications and Networks (ICCCN)*; IEEE: Piscataway, NJ, USA, 2025. <https://doi.org/10.1109/ICCCN65249.2025.11133832>.
57. Siddiqua, M.; Yousuf, N. A Comparative Analysis of RSA, ECDSA, and ED25519 in SSH Public Key Authentication. In *Proceedings of the International Conference on Computer and Information Technology (ICCIT 2025)*, Tabuk, Saudi Arabia, 13–14 April 2025.
58. Zhang, S.; Wang, S.; Chen, L.; Li, D.; Liu, B. Your Shield is My Sword: A Persistent Denial-of-Service Attack via the Reuse of Unvalidated Caches in DNSSEC Validation. In *Proceedings of the 34th USENIX Security Symposium (USENIX Security 25)*, Seattle, WA, USA, 13–15 August 2025.
59. Daniluk, C.A.; Nosyk, Y.; Duda, A.; Korczyński, M. Zeros Are Heroes: NSEC3 Parameter Settings in the Wild. In *Proceedings of the 2024 ACM Internet Measurement Conference (IMC '24)*; Association for Computing Machinery: New York, NY, USA, 2024. <https://doi.org/10.1145/3646547.3689017>.
60. Wu, D.; Zhang, Y.; Liu, B.; Li, X.; Alowaisheq, E.; Duan, H. Exploring and Analyzing Cross Layer DoS Attack Against UDP-based Services on Linux. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25)*; Association for Computing Machinery: New York, NY, USA, 2025. <https://doi.org/10.1145/3719027.3744878>.
61. Fall, A.A. SoK: Systematizing Hybrid Strategies for the Transition to Post-Quantum Cryptography. Cryptology ePrint Archive, Paper 2025/2052. 2025. Available online: <https://eprint.iacr.org/2025/2052> (accessed on 22 April 2026).
62. Benčina, B.; Dowling, B.; Maram, V.; Xagawa, K. Post-Quantum Cryptographic Analysis of SSH. In *Proceedings of the 46th IEEE Symposium on Security and Privacy (S&P)*; IEEE: Piscataway, NJ, USA, 2025; pp. 595–613. <https://doi.org/10.1109/SP61157.2025.00126>.
63. Lee, J.; Hoque, S.; Aydeger, A.; Zeydan, E. Quantum-Resistant Domain Name System: A Comprehensive System-Level Study. *arXiv* **2025**, arXiv:2506.19943. <http://arxiv.org/abs/2506.19943>.
64. Rawat, A.S.; Jhanwar, M.P. Quantum-Safe Signatureless DNSSEC. ePrint 2024/1319. 2024. Available online: <https://eprint.iacr.org/2024/1319.pdf> (accessed on 29 April 2026).
65. IETF PQUIP WG. Hybrid Signature Spectrums. Internet-Draft, draft-ietf-pquip-hybrid-signature-spectrums-07, 2025. Available online: <https://datatracker.ietf.org/doc/draft-ietf-pquip-hybrid-signature-spectrums/07/> (accessed on 15 May 2026).
66. Cybersecurity and Infrastructure Security Agency. *Post-Quantum Considerations for Operational Technology*; Technical Report; Cybersecurity and Infrastructure Security Agency: Washington, DC, USA, 2024.
67. NVD. CVE-2024-30172: Bouncy Castle Ed25519 Verification Infinite-Loop Vulnerability. National Vulnerability Database, 2024. Fixed in Bouncy Castle Java 1.78. Available online: <https://nvd.nist.gov/vuln/detail/CVE-2024-30172> (accessed on 6 May 2026).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.