

Post-Quantum Command Authentication for Unmanned Systems: A KpqC Feasibility Study under Real-Time and Bandwidth Constraints

Yulim Hyoung¹[0009–0009–7530–6497], Daeun Lim¹[0009–0009–4867–8527],
Jiwon Bang¹[0009000654531722], Doyun Park¹[0009–0002–2201–6161],
Hagyeong Kim¹[0009–0008–5840–5517], and Hwajeong Seo¹[0000–0003–0069–9061]

Hansung University, Seoul, South Korea
yulim4hyoung@gmail.com lima050627@gmail.com
bgjiiwOn754@gmail.com rhcp030418@gmail.com
kimhaha4420@gmail.com hwajeong84@gmail.com

Abstract. The command and authentication channels of unmanned systems (UAVs and UGVs) still rely on classical public-key cryptography, which large-scale quantum computers threaten. Migrating these channels to post-quantum cryptography (PQC) is therefore imperative, yet PQC keys and signatures are substantially larger, and sometimes slower, than their classical counterparts, while command links are tightly constrained in message rate, maximum transmission unit (MTU), latency deadline, and bandwidth. Whether a given PQC algorithm fits such a budget, and how to mitigate it when it does not, remains an open and practically important question. This paper studies the application of Korean PQC (KpqC: HAETAE, AIMer, SMAUG-T, NTRU+) to the authentication and key-establishment path of ROS2/DDS-Security, framed as a command-channel size–latency budget problem. We keep KpqC as the subject of study while using the NIST standards (ML-KEM, ML-DSA, SLH-DSA) as a familiar yardstick and classical ECDSA/ECDH as the migration baseline, giving a classical→NIST→KpqC three-way comparison on a single, consistent testbed. We define a feasibility envelope that maps each algorithm to the channel conditions it can meet, and we validate it with authentication-handshake measurements over emulated constrained links on a two-node testbed. Our results yield an application guideline indicating which KpqC candidate is admissible for which command-channel budget.

Keywords: Post-quantum cryptography · KpqC · Unmanned systems · DDS-Security · Command authentication · Feasibility analysis.

1 Introduction

Once large-scale quantum computers become practical, Shor’s algorithm will undermine the RSA and elliptic-curve schemes that secure today’s public-key infrastructure [19]. The “harvest-now, decrypt-later” threat, capturing traffic now

to decrypt it once such machines exist, makes the transition especially pressing for long-lived, confidentiality-critical national and defense systems [12]. NIST has responded by standardizing ML-KEM [14], ML-DSA [13], and SLH-DSA [15], and Korea has pursued its own standardization through the KpqC effort, selecting HAETAE and AIMer for signatures and SMAUG-T and NTRU+ for key encapsulation [10]. In a defense and procurement setting, moving to a domestic standard is also a matter of cryptographic sovereignty.

The command and authentication channels of unmanned systems are a natural focus of this transition. MAVLink, a common protocol for uninhabited aircraft, offers message authentication only as an option in version 2, a pre-shared key with a truncated HMAC, and none at all in version 1, so it carries the familiar weaknesses of symmetric keying [11]. ROS2 and its security layer, DDS-Security, instead rely on X.509 certificates and Diffie–Hellman over a NIST curve for mutual authentication and key agreement [16]; this authentication and key-establishment step is exactly where a quantum adversary would strike. Because a forged, replayed, or injected command can hand an attacker control of the platform, raising the strength of this step is a priority.

Substituting post-quantum primitives is not, however, guaranteed to work on such channels. Post-quantum keys, signatures, and ciphertexts are several times to an order of magnitude larger than their classical counterparts, and some are slower to compute. Command and telemetry links, meanwhile, are tightly bounded in rate, packet size, deadline, and bandwidth: a typical UAV telemetry radio runs at tens of kilobits per second [1], a MAVLink v2 frame leaves only about 250 bytes of payload [11], and a low-rate satellite link allows only a few hundred bytes per message with delays measured in seconds [8]. Under these limits a large signature can turn into a real bottleneck, forcing fragmentation, missing a deadline, or saturating the link. Whether a given scheme fits a channel’s budget, and what to do when it does not, is therefore an open and practical question.

Recent studies applying NIST post-quantum schemes to drones are appearing quickly [18,3,5], yet to our knowledge none applies and evaluates KpqC on the real-time command channels of unmanned systems, and most report single-device or wired benchmarks rather than treating the problem as a channel budget measured over a constrained link. We address this gap. We keep KpqC at the center of the study, use the NIST standards as a familiar point of reference, and use classical ECDSA/ECDH as the starting point, giving a classical→NIST→KpqC comparison.

This paper makes the following contributions.

- To our knowledge, the first application and evaluation of KpqC on the authentication and key-establishment path of ROS2/DDS-Security: key establishment is demonstrated end to end, and we report the provider integration limit that currently bounds the signature path (C1).
- A budget model that, from a channel’s message size, rate, deadline, and bandwidth, decides which schemes are admissible, the feasibility envelope, which we treat as the core result (C2).

- A three-way dataset of size and speed for classical, NIST, and KpqC measured on one consistent platform, read from the perspective of command rate and packet size (C3).
- Handshake measurements on a two-node testbed over emulated constrained links, which check the envelope in a networked setting (C4).

What we build is a controlled testbed, and we keep our claims to what it can show: performance and feasibility in this setting. The cost of a cryptographic operation, however, is inherent to the algorithm and the platform, so the figures carry over to any deployment that uses the same primitives, whether or not the target here is a fielded system, an argument we make explicit in Section 4.

The rest of the paper is organized as follows. Section 2 covers post-quantum and KpqC background together with command-channel security and related work. Section 3 presents the threat model, where KpqC is applied, the choice of algorithms, the budget model, and the testbed. Section 4 reports the three-way results, the budget comparison, the handshake measurements, and the limitations, and Section 5 concludes.

2 Background and Related Work

2.1 Background

Post-quantum cryptography. A large quantum computer running Shor’s algorithm [19] would break RSA and elliptic-curve cryptography, so standards bodies have moved to replace them. NIST finalized three schemes in 2024: the lattice-based key-encapsulation mechanism ML-KEM [14], the lattice-based signature ML-DSA [13], and the hash-based signature SLH-DSA [15]. These families trade off differently, lattice schemes are compact and fast but rest on structured assumptions, while hash-based schemes are conservative but produce large signatures, and that trade-off is what matters for a tight channel. Korea’s KpqC process selected domestic counterparts: the signatures HAETAE (lattice) and AIMer (built on a symmetric primitive), and the key encapsulation mechanisms SMAUG-T and NTRU+ [10]. In this paper we study these KpqC schemes as the main subject and pair each with the closest NIST standard as a reference, HAETAE with ML-DSA, AIMer with SLH-DSA, and SMAUG-T/NTRU+ with ML-KEM, while classical ECDSA/ECDH serves as the starting point. Prior implementations of the KpqC schemes reported in earlier work [4,6,7] provide a reference point for the parameter sets we measure.

Command-channel security. MAVLink v2 authenticates a message with a 13-byte field derived from a pre-shared key (v1 has no authentication at all), which limits how it scales as keys must be distributed and revoked [11]; moreover, a frame holds at most about 250 payload bytes, too little for a public-key signature. In ROS2, DDS-Security performs certificate-based mutual authentication with an ECDSA or RSA signature and derives a shared key by Diffie–Hellman, after which the traffic itself is protected symmetrically [16]. The consequence matters

for us: only the signatures and the key agreement are quantum-vulnerable and thus the natural targets for KpqC, whereas the steady flow of commands, protected by a symmetric key, is largely unaffected. The impact of a post-quantum change therefore concentrates in the session setup.

Real-time and bandwidth limits. The links carrying these sessions are constrained in different ways, but every constraint reduces to the same three quantities: bandwidth, message size, and deadline. A telemetry radio offers tens of kilobits per second, a frame of about 250 bytes, and a command deadline of hundreds of milliseconds [1,11]. A local control loop offers gigabit bandwidth, a 1500-byte frame, and a deadline of one or two milliseconds at hundreds of hertz [17]. A satellite short-burst link offers a few kilobits per second, a message of a few hundred bytes, and a deadline of seconds [8]. These three budgets, rather than raw compute, decide whether a large post-quantum parameter is usable, and they are the axes of the model in Section 3.4.

2.2 Related Work

Post-quantum ROS2/DDS. The closest work integrates NIST ML-KEM and ML-DSA into ROS2/DDS-Security and measures them on an embedded board [18]. It targets the NIST standards in a wired, single-device setting; we differ by studying KpqC, by measuring over a bandwidth-limited link, and by framing the question as a channel budget.

Post-quantum MAVLink. Other work adds NIST post-quantum schemes to MAVLink; one implementation observes a relay truncating signatures that spill across a frame boundary [5,2]. That symptom is exactly the bandwidth pressure our budget model sets out to capture.

Budget and mitigation. For vehicle-to-vehicle communication, hybrid classical-plus-post-quantum authentication and certificate caching have been proposed to absorb the added cost under a strict budget [22]; related work schedules post-quantum operations against real-time deadlines in simulation [3]. These give us a template for the budget and mitigation view, which we move to unmanned-system command channels and to KpqC.

KpqC in practice. KpqC has been integrated into TLS and benchmarked [20], but for general-purpose TLS rather than a command channel, so it serves us mainly as a methodological reference; a separate comparison of KpqC against NIST schemes on a flight-controller-class microcontroller offers a useful basis for choosing among the candidates [21]. Our contribution is to lift such measurements onto a real protocol and a constrained link.

Positioning. Table 1 summarizes the comparison. Prior work covers each axis separately, NIST schemes on a real protocol, KpqC in general-purpose TLS or at the primitive level, and budget or mitigation views in simulation or V2V,

Table 1. Related work at a glance. ✓: addressed; △: partial or indirect; −: not addressed. “Constrained link” means measurement over a bandwidth- or delay-limited channel rather than a wired or single-host setup; “budget model” means an explicit size/rate/deadline admissibility criterion.

Work	Schemes	Protocol / target	Constr. link	Budget	Mitig.
[18]	NIST	ROS2/DDS-Security	−	−	−
[5,2]	NIST	MAVLink	△	−	−
[22]	NIST	V2V	△	✓	✓
[3]	NIST	UAV swarm (sim.)	−	△	△
[20]	KpqC	TLS/X.509 (general)	−	−	−
[21]	KpqC, NIST	Cortex-M4 primitives	−	−	△
This work	classical, NIST, KpqC	ROS2/DDS-Security (key est.)	✓	✓	△

but no single study combines them. The closest work [18] integrates NIST ML-KEM/ML-DSA into DDS and reports time and size on an embedded device; we build on the same target and the same metrics. Our study extends that setting in three respects, which Table 1 makes explicit: it turns to the KpqC family, it measures over a bandwidth-limited two-node link, and it asks not only what each scheme costs but whether that cost fits a given channel’s budget. In short, the novelty is the intersection of KpqC, a real-time/bandwidth budget, and ROS2/DDS command authentication. Putting post-quantum cryptography on an unmanned platform is by now well trodden [9]; doing so for KpqC, as a channel budget, on this protocol is not, and that is the gap we fill.

3 Method

3.1 Threat Model and Requirements

We consider an adversary that can inject, forge, or replay messages on the command channel of an unmanned system, with the aim of issuing an unauthorized command or replaying a captured one [9]. The adversary is classically bounded today but may record traffic now and decrypt it once a quantum computer becomes available. We therefore require authenticity and freshness of every command, and confidentiality of command content, against such an adversary. Two operational constraints shape any solution. The first is a real-time deadline: a command that authenticates too late is useless, so the authentication cost must fit within the round-trip budget of the channel. The second is a bandwidth and packet budget: command links are narrow, and a single frame often carries only a few hundred bytes, so large keys or signatures either overflow the frame or saturate the link. We draw these requirements from public protocol specifications rather than from any particular deployment. Section 3.4 turns these two constraints into a per-channel test.

3.2 Where KpqC Is Applied

The natural insertion point is the authentication and key-establishment step. In DDS-Security, participant authentication uses X.509 certificates and a Diffie–

Hellman key agreement over a NIST curve [16]. We replace the key agreement with a post-quantum KEM and the certificate signatures with a post-quantum signature, and leave the steady-state command traffic to the symmetric keys that this step establishes. The post-quantum cost is thus paid once per session rather than on every command, a structure we return to in the mitigation discussion. To measure these substitution points directly we map them onto a TLS 1.3 handshake, which shares the same certificate, signature, and key-agreement structure as the DDS-Security handshake [20].

3.3 Algorithm Selection and Comparison Design

We keep KpqC at the center and compare each KpqC scheme with its closest NIST standard and with a classical baseline, so that every result reads as a classical→NIST→KpqC progression. For signatures, HAETAE is paired with ML-DSA, since both are lattice schemes, and AIMer with SLH-DSA, since both rest on conservative, non-lattice assumptions; Falcon is included as an additional NIST lattice reference. For key encapsulation, SMAUG-T and NTRU+ are paired with ML-KEM. The classical baseline is ECDSA and ECDH over the P-256 curve. Every scheme runs through the same harness, so the three-way comparison is internally consistent.

3.4 Budget Model (Feasibility Envelope)

The core of the method is a per-channel admissibility test that operationalizes the two constraints of Section 3.1: the deadline becomes D , and the bandwidth and packet budget become b and m . A channel is described by its message rate f , bandwidth B , maximum transmission unit m , and round-trip deadline D . From the rate and bandwidth we derive a per-message byte budget $b = \alpha B / (8f)$: at rate f the link carries $B / (8f)$ bytes per message, and we allow a fraction $\alpha = 0.5$ of that to be spent on authentication. Each scheme is reduced to two numbers. Its on-wire size s is the signature length for a signature scheme, and $\text{pk} + \text{ct}$ (public key plus ciphertext) for a KEM. Its operation latency t is $\text{sign} + \text{verify}$ for a signature scheme, and $\text{keygen} + \text{encap} + \text{decap}$ for a KEM. A scheme is admissible for the channel when it passes all three of the following tests: the size fits the byte budget ($s \leq b$), the size fits a single frame ($s \leq m$, otherwise fragmentation is required), and the latency fits the deadline ($t \leq D$). Collecting these verdicts over all schemes and channels gives the feasibility envelope, the central artifact of this study: a table that states, for each channel, which schemes fit within a single message and a deadline and which do not.

3.5 Testbed and Implementation

All primitive measurements are performed on one platform, using `liboqs_KpqC` for the post-quantum schemes and OpenSSL for the classical baseline. The authentication handshake is realized through the `oqs-provider_KpqC` provider as

a TLS 1.3 handshake, first over loopback and then between two endpoints placed in separate network namespaces joined by a virtual link, over which constrained channel profiles are shaped with `netem`. This two-node arrangement stands in for a commander and a vehicle without a fielded radio; its limits are stated in Section 4.5, and the measurement details, including warm-up and median reporting, are given with the results in Section 4.1.

4 Experiment

4.1 Setup

All primitives are measured through a single harness built on `liboqs_KpqC` with the `oqs-provider_KpqC` OpenSSL 3 provider, so that classical, NIST, and KpqC schemes are exercised through the same code path. Because AVX2-optimized implementations of the KpqC algorithms are not yet available, we build the library in its portable C configuration and, for a fair comparison, disable the optimized code paths for the NIST schemes as well; all reported latencies are therefore portable-C figures on the same x86-64 platform under Ubuntu 26.04 LTS, an Intel Core i5-1135G7 laptop with OpenSSL 3.5.

Primitive latencies are reported as medians over 10^2 iterations after a short warm-up. The authentication handshake is measured in two ways.

- *TLS proxy*. A TLS 1.3 handshake standing in for the DDS-Security handshake, run on the loopback interface of the primitive-benchmark host.
- *Two-node testbed*. A commander and a vehicle endpoint placed in separate Linux network namespaces joined by a virtual link, on which we shape three channel profiles with `netem`. The profiles approximate a low-rate satellite link (SATCOM, 2.4 kbps, 500 ms delay), a UAV telemetry radio (UAV, 57.6 kbps, 40 ms delay), and a near-unconstrained wired link (clear).

The primitive benchmarks, the feasibility envelope, and the TLS-proxy handshake are taken on one host, whereas the two-node handshake is taken on a separate Ubuntu host running OpenSSL 3.0. Absolute handshake times therefore differ between the two settings, and we compare only within each.

4.2 Three-way Primitive Performance

Signatures Tables 2 and 3 report size and speed for signatures and key encapsulation across the classical, NIST, and KpqC families; the size columns are visualized in Figure 1 and the latencies in Figure 2. The classical baseline is, as expected, both the smallest and among the fastest, which is why its replacement is costly. Among the lattice signatures, KpqC’s HAETAE is competitive with ML-DSA: its public keys and signatures are smaller at matched security levels (for example 1474 B versus 2420 B at level 2/44), while sign and verify stay in the sub-millisecond to low-millisecond range. Falcon has the most compact signatures of the lattice set but the slowest key generation by an order of magnitude.

Table 2. Signature schemes: sizes (bytes) and portable-C latencies (ms), medians over 10^2 iterations. SLH-DSA rows use the SPHINCS+-SHA2-*simple* set.

Scheme	Class	pk	sig	keygen	sign	verify
ECDSA-P256	classical	65	71	0.014	0.024	0.062
ML-DSA-44	NIST	1312	2420	0.139	0.237	0.072
ML-DSA-65	NIST	1952	3309	0.128	0.422	0.125
ML-DSA-87	NIST	2592	4627	0.200	0.522	0.206
Falcon-512	NIST	897	752	12.84	4.55	0.035
Falcon-1024	NIST	1793	1462	37.87	10.28	0.069
SLH-DSA-128f	NIST	32	17088	0.79	18.57	1.09
SLH-DSA-128s	NIST	32	7856	50.44	383.9	0.37
SLH-DSA-256f	NIST	64	49856	3.00	65.85	1.73
SLH-DSA-256s	NIST	64	29792	47.94	651.6	0.88
HAETA-2	KpqC	992	1474	0.273	0.388	0.052
HAETA-3	KpqC	1472	2349	0.535	0.343	0.089
HAETA-5	KpqC	2080	2948	0.629	2.960	0.109
AIMer-128f	KpqC	32	5888	0.023	1.080	1.001
AIMer-128s	KpqC	32	4160	0.023	8.455	8.361

For the conservative, non-lattice signatures the contrast is sharper. KpqC’s AIMer produces signatures of 4–6 kB and signs in about one to eight milliseconds, whereas the hash-based SLH-DSA pays for its minimal assumptions with signatures from 7.9 kB up to 49 kB and signing times that reach hundreds of milliseconds for the small-signature variants. Where a conservative signature is required, AIMer is markedly lighter on both size and speed.

Table 3. KEM schemes: sizes (bytes) and portable-C latencies (μ s), medians over 10^2 iterations.

Scheme	Class	pk	ct	keygen	encap	decap
ECDH-P256	classical	65	65	13.5	102.6	50.8
ML-KEM-512	NIST	800	768	19.3	22.9	27.2
ML-KEM-768	NIST	1184	1088	31.8	33.0	40.2
ML-KEM-1024	NIST	1568	1568	50.0	49.7	59.7
SMAUG-T1	KpqC	672	672	24.4	22.5	30.3
SMAUG-T3	KpqC	1088	992	48.4	46.1	61.3
SMAUG-T5	KpqC	1440	1376	78.0	73.5	90.1
NTRU+-576	KpqC	864	864	42.2	25.5	32.1
NTRU+-768	KpqC	1152	1152	63.7	36.2	45.7
NTRU+-864	KpqC	1296	1296	67.3	38.8	54.4
NTRU+-1152	KpqC	1728	1728	98.8	54.2	70.4

KEMs The KEMs are uniformly inexpensive in time, tens of microseconds per operation, so key encapsulation is dominated by size, not computation. Here the KpqC entries are attractive: SMAUG-T1 has the smallest public key and ciphertext of the whole set (672 B each), below ML-KEM-512, and SMAUG-T3 and the NTRU+ family sit close to ML-KEM-768. This compactness is what later decides the outcome on a bandwidth-bound link.

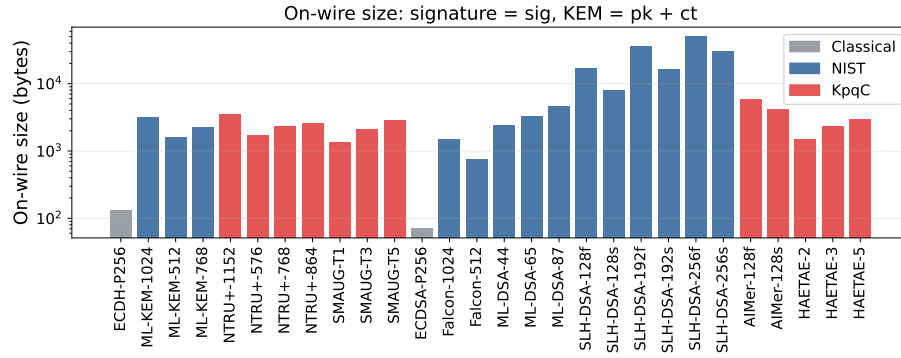


Fig. 1. On-wire size per scheme (log scale): signature bytes for signatures, public key + ciphertext for KEMs, grouped as classical / NIST / KpqC.

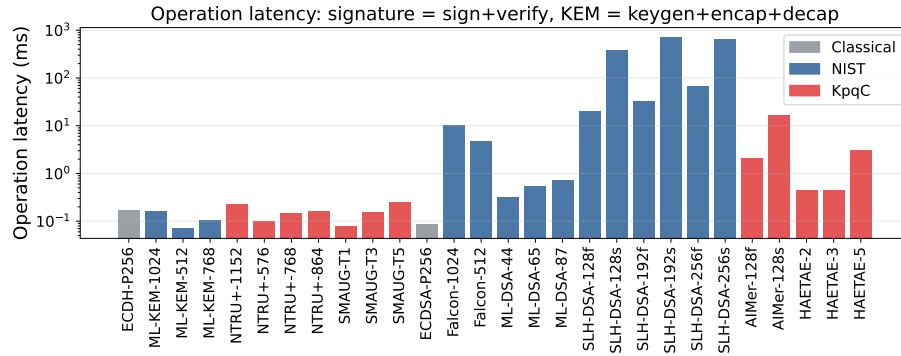


Fig. 2. Portable-C operation latency per scheme (log scale): sign/verify for signatures, keygen/encap/decap for KEMs, grouped as classical / NIST / KpqC.

4.3 Feasibility Envelope

The budget model turns these numbers into a per-channel admissibility test. For each channel we derive a per-message byte budget from its rate and bandwidth and compare it, together with the MTU and the round-trip deadline, against each scheme’s on-wire material and operation latency. Figure 3 summarizes the outcome over three representative channels: a MAVLink-class UAV command channel (MTU 253 B), a wired ROS2/DDS channel (MTU 1500 B), and a low-rate SATCOM link (MTU 340 B). Table 4 lists their parameters.

Table 4. Command-channel budget parameters. Values marked † are estimates; the rest follow public specifications (MAVLink v2 and SiK radio [11,1], `ros2_control` [17], Iridium SBD [8]). The per-message byte budget is $b = \alpha B / (8f)$ with $\alpha = 0.5$ (an authentication-overhead fraction).

Channel	rate (Hz)	bandwidth	MTU (B)	budget (B)	cmd deadline	hs deadline
MAVLink-UAV	10	57.6 kbps	253	360	500 ms†	5 s
ROS2-DDS	500	1 Gbps†	1500	125000	2 ms	500 ms
SATCOM-SBD	0.1	2.4 kbps	340	1500	20 s	60 s

Two readings stand out.

- *Constrained RF links.* No post-quantum scheme fits within a single message: every NIST and KpqC signature and key exchange exceeds the MTU and must be fragmented, and only the classical baseline passes. This is a direct, quantified statement of the tension this paper studies, and the motivation for the mitigations of Section 4.5.
- *Wired ROS2/DDS channel.* The picture changes and KpqC is the discriminating case. HAETAE-2 and SMAUG-T1 pass where their NIST counterparts do not: ML-DSA exceeds the MTU, and Falcon misses the latency bound in this portable-C setting. The smallest KpqC schemes are thus the first post-quantum options to become admissible as the channel loosens.

4.4 Authentication Handshake

The envelope above judges per-message signing. Authentication and key establishment, however, are a one-time session cost, and we measure them directly. As a proxy for the DDS-Security handshake, which rests on the same X.509 certificate, signature, and key-agreement primitives, we run a TLS 1.3 handshake, first over loopback and then over the two-node testbed, and record the on-wire bytes and the completion time.

Certificates A first result concerns the certificate itself. When KpqC signature keys are used to issue the identity and participant certificates, HAETAE yields consistently smaller certificates than ML-DSA at each level (server certificate

Feasibility envelope (PASS/FAIL)

ECDH-P256	PASS	PASS	PASS
ECDSA-P256	PASS	PASS	PASS
AIMER-128f	FAIL	FAIL	FAIL
AIMER-128s	FAIL	FAIL	FAIL
HAETAE-2	FAIL	PASS	FAIL
HAETAE-3	FAIL	FAIL	FAIL
HAETAE-5	FAIL	FAIL	FAIL
NTRU-Plus-KEM-1152	FAIL	FAIL	FAIL
NTRU-Plus-KEM-576	FAIL	FAIL	FAIL
NTRU-Plus-KEM-768	FAIL	FAIL	FAIL
NTRU-Plus-KEM-864	FAIL	FAIL	FAIL
SMAUG-T1	FAIL	PASS	FAIL
SMAUG-T3	FAIL	FAIL	FAIL
SMAUG-T5	FAIL	FAIL	FAIL
Falcon-1024	FAIL	FAIL	FAIL
Falcon-512	FAIL	FAIL	FAIL
ML-DSA-44	FAIL	FAIL	FAIL
ML-DSA-65	FAIL	FAIL	FAIL
ML-DSA-87	FAIL	FAIL	FAIL
ML-KEM-1024	FAIL	FAIL	FAIL
ML-KEM-512	FAIL	FAIL	FAIL
ML-KEM-768	FAIL	FAIL	FAIL
SPHINCS+-SHA2-128f-simple	FAIL	FAIL	FAIL
SPHINCS+-SHA2-128s-simple	FAIL	FAIL	FAIL
SPHINCS+-SHA2-192f-simple	FAIL	FAIL	FAIL
SPHINCS+-SHA2-192s-simple	FAIL	FAIL	FAIL
SPHINCS+-SHA2-256f-simple	FAIL	FAIL	FAIL
SPHINCS+-SHA2-256s-simple	FAIL	FAIL	FAIL

MAVLink-UAV-10Hz ROS2-DDS-500Hz SATCOM-SBD-0.1Hz

Fig. 3. Feasibility envelope (PASS/FAIL) of each scheme against three channel budgets. Green marks admissibility within a single message under the channel's MTU, byte budget, and deadline.

2697 B for HAETAE-2 against 3972 B for ML-DSA-44, and 5259 B against 7458 B at the top level)¹, which matters directly for the certificate exchange during the handshake. We note, however, a practical integration limit: in the provider build available to us the KpqC signature algorithms are exposed for X.509 issuance but not yet wired as TLS signature schemes, so a KpqC-signed certificate could not be used for server authentication in the handshake. We therefore hold the certificate fixed and vary the key-exchange group, which isolates the effect of the KEM.

Two-node handshake Table 5 reports the two-node handshake over the clear and UAV profiles. On the clear link every handshake completes in about 22 ms, and the schemes are indistinguishable: at this bandwidth the time is dominated by process and provider setup, not by the key exchange, so the on-wire bytes are the meaningful quantity. On the constrained UAV link the bytes translate into time, and the schemes separate: the classical P-256 exchange is fastest at 639 ms, and among the post-quantum groups SMAUG-T1 completes in 820 ms, roughly 120 ms ahead of ML-KEM-768 at 943 ms, tracking its smaller handshake footprint. The envelope’s size-based prediction thus shows up as a real latency difference once the link is bandwidth-bound.

Table 5. Two-node handshake over emulated links (ECDSA certificate held fixed, key-exchange group varied). Handshake time in ms (median); on-wire bytes are server→client (read) and client→server (written).

Group	Class	clear (ms)	UAV (ms)	wire rd/wr (B)
P-256	classical	20.9	639	1018 / 320
ML-KEM-768	NIST	22.3	943	2041 / 1431
SMAUG-T1	KpqC	22.4	820	1625 / 919
SMAUG-T3	KpqC	22.3	923	1945 / 1335
NTRU+-768	KpqC	22.2	941	2105 / 1399

DDS-Security integration We scope the integration claim to what the measurements support. The TLS-as-proxy handshake exercises the same authenticated key-establishment primitives that DDS-Security uses for participant authentication, and the two-node result shows that KpqC key exchange completes successfully over constrained links. A full substitution inside a running ROS2/DDS-Security stack, in particular using KpqC *signatures* for participant certificates, is blocked by the provider integration limit noted above and is left as future work once KpqC signature schemes are available as TLS signature algorithms.

¹ Certificate DER sizes vary by a byte or two with the serial-number encoding; we report to the nearest byte.

4.5 Discussion

Mitigations The envelope points to three responses when a scheme overflows a channel budget; we frame them here and leave their measurement to future work.

- *Fragmentation.* An oversized signature or key is split across MTU-sized frames, and the envelope’s byte figures give the count: on the MAVLink channel a HAETAE-2 signature needs six frames and ML-DSA-44 ten. Beyond airtime (1474 B is about 205 ms at 57.6 kbps), the loss of any one fragment invalidates the signature, so fragmentation suits low-loss links and infrequent commands.
- *Hybrid operation.* A classical and a post-quantum scheme run together. The increment is small, ECDSA-P256 adds 71 B and under 0.1 ms to HAETAE-2, but hybrid does not restore admissibility on a channel the post-quantum scheme already overflows; its value is assurance during migration, not budget.
- *Authentication caching.* The handshake results expose a structure that avoids the per-message budget: the post-quantum cost falls once on session setup, after which commands carry only a symmetric tag (16 B for AES-GCM, or the 13 B field of MAVLink v2) that fits any frame. A SMAUG-T1 handshake of 820 ms over the UAV link, spread over a ten-minute sortie at 10 Hz, amounts to 0.14 ms per command. A channel that cannot afford a post-quantum signature per command can afford one per session, the regime in which the compact KpqC schemes become practical.

Limitations Our claims are bounded to this testbed; the cryptographic-cost figures, being intrinsic to the algorithms, transfer to any deployment of the same primitives. Four caveats apply. First, latencies are portable-C figures from a single library, taken uniformly for NIST and KpqC: they overstate schemes with mature vectorized code and may mix reference-code maturity with algorithm cost, so latency-driven failures are conservative, whereas the size-driven results that decide most of the envelope are unaffected. Second, the channel parameters are specification-derived estimates (Table 4). Third, the two-node handshake is emulated with `netem`, which captures bandwidth, delay, and loss but not radio-layer effects, and the TLS 1.3 proxy differs from the DDS-Security handshake in transport and message sequencing, so the absolute times and bytes of Table 5 are not those of a fielded deployment. Finally, the KpqC-signature integration gap bounds the DDS-Security claim to the key-establishment path; it lies in the TLS proxy, not in DDS, and closes with a provider update.

5 Conclusion

We studied the application of KpqC to the authentication and key-establishment path of unmanned-system command channels, framed as a size and latency budget problem, and evaluated it against NIST and classical baselines on a single,

consistent testbed. Two findings organize the result. On tightly bounded RF links no post-quantum scheme fits within a single message, so a post-quantum transition on these links cannot be a drop-in substitution of primitives: per-command signing requires one of the mitigations we outline, most directly moving the post-quantum cost to the session handshake. Since such links are the common case for unmanned systems, and since an adversary can already record today’s traffic for later decryption, the question for these platforms is how to make this change, not whether. On a wired channel, by contrast, the most compact KpqC schemes, HAETAE-2 for signatures and SMAUG-T1 for key exchange, are the first post-quantum options to become admissible, ahead of their NIST counterparts. The same ordering appears in the two-node handshake, where over a constrained UAV link SMAUG-T1 completes ahead of ML-KEM-768 by tracking its smaller on-wire footprint. The feasibility envelope thus indicates which KpqC candidate is admissible under which channel budget, and it doubles as an application guideline for post-quantum command authentication: use the compact KpqC schemes at the session handshake and let symmetric keys carry the steady-state command traffic. Future work includes full DDS-Security integration once KpqC signatures are available as TLS signature schemes, measurement of the mitigation techniques, evaluation over live constrained links, and validation with real interface-control parameters.

References

1. ArduPilot: SiK Radio: Advanced Configuration and Technical Information. <https://ardupilot.org/copter/docs/common-3dr-radio-advanced-configuration-and-technical-information.html> (2024), last accessed 31 Aug 2026
2. Augusto, C.: Post-Quantum Authentication for MAVLink C2: Found a Relay Problem Nobody Talks About. ArduPilot Discourse, <https://discuss.ardupilot.org/t/144270>; implementation at <https://github.com/cleitonaugusto/CleitonQ> (2026), last accessed 31 Aug 2026
3. Aydeger, A., Carvalho, M.: Real-Time Cryptographic Agility for Autonomous UAV Swarms: A Performance-Driven Approach to Post-Quantum Migration. In: 2026 IEEE 23rd Consumer Communications & Networking Conference (CCNC) (2026). <https://doi.org/10.1109/CCNC65079.2026.11366545>
4. Cho, S., et al.: A Memory-Efficient and Assembly-Optimized Implementation of NTRU+. IACR Cryptology ePrint Archive, Paper 2026/1410 (2026), <https://eprint.iacr.org/2026/1410>
5. Epure, L.I.: Post-Quantum Cryptography for Military UAV Communication Systems: A Comprehensive Framework for Quantum-Resistant Security in MAVLink-Based Unmanned Aerial Vehicles. Authorea preprint (2025). <https://doi.org/10.22541/au.176281850.08964861/v1>
6. Hyoung, Y.L., et al.: An Optimized Native Rust Implementation of the KpqC Algorithms. IACR Cryptology ePrint Archive, Paper 2026/1420 (2026), <https://eprint.iacr.org/2026/1420>
7. Hyoung, Y.L., et al.: Memory-Efficient Implementation of SMAUG-T and HAETAE. IACR Cryptology ePrint Archive, Paper 2026/442 (2026), <https://eprint.iacr.org/2026/442>

8. Iridium Communications Inc.: Iridium 9602 SBD Transceiver Developer's Guide. Iridium Short Burst Data Service documentation (2012), maximum MO message 340 bytes, MT message 270 bytes
9. Khan, M.A., et al.: Future-Proofing Security for UAVs With Post-Quantum Cryptography: A Review. *IEEE Open Journal of the Communications Society* **5**, 6849–6871 (2024). <https://doi.org/10.1109/OJCOMS.2024.3486649>
10. KpqC: Korean Post-Quantum Cryptography Competition: Final Selection of HAETAE, AIMer, SMAUG-T, and NTRU+. <https://kpqc.cryptolab.co.kr/> (2025), last accessed 31 Aug 2026
11. MAVLink: MAVLink Developer Guide: Message Signing and Packet Serialization. <https://mavlink.io/> (2024), last accessed 31 Aug 2026
12. Mosca, M.: Cybersecurity in an Era with Quantum Computers: Will We Be Ready? *IEEE Security & Privacy* **16**(5), 38–41 (2018). <https://doi.org/10.1109/MSP.2018.3761723>
13. National Institute of Standards and Technology: Module-Lattice-Based Digital Signature Standard. FIPS 204 (2024). <https://doi.org/10.6028/NIST.FIPS.204>
14. National Institute of Standards and Technology: Module-Lattice-Based Key-Encapsulation Mechanism Standard. FIPS 203 (2024). <https://doi.org/10.6028/NIST.FIPS.203>
15. National Institute of Standards and Technology: Stateless Hash-Based Digital Signature Standard. FIPS 205 (2024). <https://doi.org/10.6028/NIST.FIPS.205>
16. Object Management Group: DDS Security Specification, Version 1.1 (DDS:Auth:PKI-DH, DDS:Crypto:AES-GCM-GMAC). OMG formal specification, <https://www.omg.org/spec/DDS-SECURITY/1.1> (2018), last accessed 31 Aug 2026
17. ros2_control Working Group: ros2_control Documentation: Controller Manager. https://control.ros.org/rolling/doc/ros2_control/controller_manager/doc/userdoc.html (2026), last accessed 31 Aug 2026
18. Safdar, U., et al.: Enhancing ROS 2 Security with Standardized Post-Quantum Cryptosystems. *International Journal of Information Security* **24**(5) (2025). <https://doi.org/10.1007/s10207-025-01133-w>
19. Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing* **26**(5), 1484–1509 (1997). <https://doi.org/10.1137/S0097539795293172>
20. Sim, M., et al.: Integrating and Benchmarking KpqC in TLS/X.509. *Electronics* **14**(18), 3717 (2025). <https://doi.org/10.3390/electronics14183717>, IACR ePrint 2025/1245
21. Sim, M., et al.: Evaluating Hybrid KEM/DSA for KpqC and NIST PQC on ARM Cortex-M4. IACR Cryptology ePrint Archive, Paper 2026/1414 (2026), <https://eprint.iacr.org/2026/1414>
22. Twardokus, G., et al.: When Cryptography Needs a Hand: Practical Post-Quantum Authentication for V2V Communications. In: Network and Distributed System Security Symposium (NDSS) (2024), IACR ePrint 2022/483