PQC-MAV

A Complete Post-Quantum Cryptographic Tunnel for UAV–GCS Communication

Comprehensive Benchmark Report

Burak Güneysu • February 2026

72 Cipher Suites · 3 Scenarios · 432 Data Points · Real Hardware

1. Executive Summary

PQC-MAV is a bump-in-the-wire post-quantum cryptographic tunnel that transparently encrypts MAVLink telemetry between a Raspberry Pi 5 drone companion computer and a Windows 11 ground-control station (GCS).

The system implements 72 cipher suites (9 KEMs × 8 SIGs × 3 AEADs) spanning NIST security levels 1, 3, and 5 across three distinct mathematical assumptions: lattice (ML-KEM), quasi-cyclic code (HQC), and binary Goppa code (McEliece).

Three complete benchmark runs were executed on February 11, 2026:

- Baseline (No DDoS): run 20260211_141627 — 72 suites, 144 JSON files
- + XGBoost Detector: run 20260211_150013 — 72 suites, 144 JSON files
- + TST Detector: run 20260211_171406 — 72 suites, 144 JSON files

Each suite undergoes a full 110-second MAVLink session with real Pixhawk SITL telemetry at 320 Hz, producing 18 categories (A–R) of metrics. Total dataset: 432 JSON files, 71/72 suites succeed per scenario.

Key findings:

1. ML-KEM dominates at every NIST level (median handshake: 14 ms)
2. McEliece handshakes are 43× slower (median: 620 ms)
3. SPHINCS+ signing is the bottleneck (642–1342 ms)
4. XGBoost detection overhead is minimal (< 9% on ML-KEM)
5. TST detection raises CPU to 89% and temp to 83°C
6. AEAD differences are negligible (46–228 µs per packet)

2. Hardware Testbed & Methodology

DRONE (uavpi)	GCS (lappy)
Raspberry Pi 5	Windows 11 x86-64
ARM Cortex-A76 (4 cores)	Python 3.11.13
3,796 MB RAM	liboqs 0.12.0
Linux 6.12.47+rpt-rpi-v8	IP: 192.168.0.101
Python 3.11.2, liboqs 0.12.0	
Power: RPi5 hwmon	
IP: 192.168.0.100	
Network: Ethernet LAN (sub-ms RTT)	

Benchmark Protocol (per suite):
  1. Drone scheduler selects next suite from 72-suite registry
  2. Sends start_proxy JSON-RPC to GCS via TCP:48080
  3. PQC handshake executes (KEM keygen → ServerHello → ClientFinish → HKDF)
  4. 110-second MAVLink data session at 320 Hz
  5. Drone sends stop_suite RPC, merges drone+GCS metrics
  6. JSON output written to logs/benchmarks/runs/{scenario}/

Three benchmarking scenarios (each runs all 72 suites):
  Scenario 1: Baseline — no DDoS detection, pure tunnel overhead
  Scenario 2: + XGBoost detector — lightweight ML model (5-pkt window, ~3s latency)
  Scenario 3: + TST detector — heavy Transformer model (400-pkt window, ~240s latency)

3. 72 Cipher Suite Registry

Suite formula: S = { (KEM_i, SIG_j, AEAD_k) | Level(KEM_i) = Level(SIG_j) }

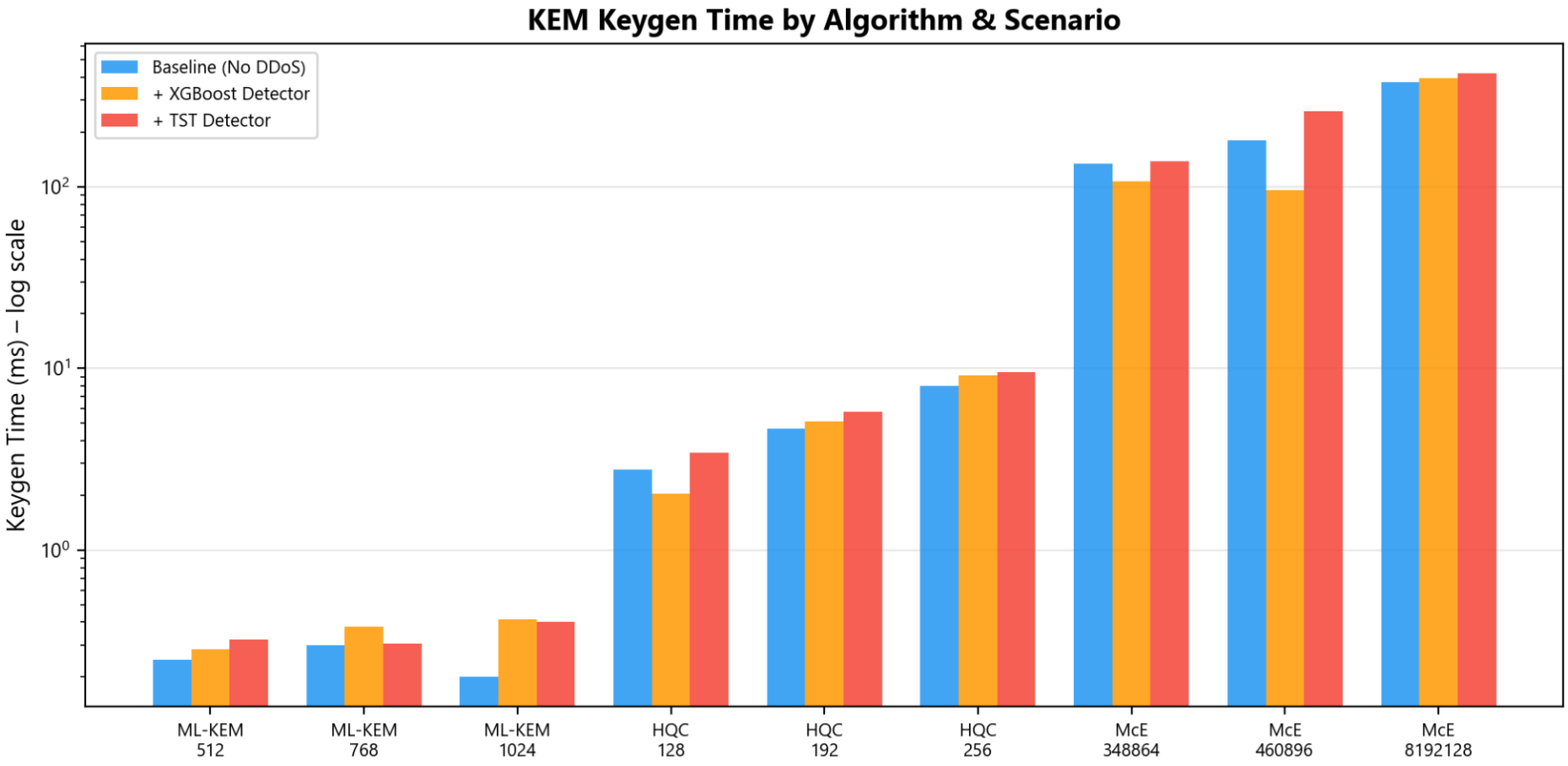
Type	Family	Variants	Levels
KEM	ML-KEM (FIPS 203)	512, 768, 1024	1, 3, 5
KEM	Classic McEliece	348864, 460896, 8192128	1, 3, 5
KEM	HQC	128, 192, 256	1, 3, 5
SIG	ML-DSA (FIPS 204)	44, 65, 87	1, 3, 5
SIG	Falcon	512, 1024	1, 5
SIG	SPHINCS+ (FIPS 205)	128s, 192s, 256s	1, 3, 5
AEAD	AES-256-GCM	—	—
AEAD	ChaCha20-Poly1305	—	—
AEAD	Ascon-128a	—	—

Level distribution:
  L1 = 3 KEMs × 3 SIGs × 3 AEADs = 27 suites   (Falcon at L1 and L5 only)
  L3 = 3 KEMs × 2 SIGs × 3 AEADs = 18 suites   (no Falcon at L3)
  L5 = 3 KEMs × 3 SIGs × 3 AEADs = 27 suites
  ──
  Total: 72 cipher suites

Mathematical diversity:
  • ML-KEM: Module-LWE (lattice)   → fastest, smallest keys
  • HQC: Quasi-cyclic codes   → moderate speed, medium keys
  • McEliece: Binary Goppa codes   → slowest keygen, huge keys (up to 1.36 MB)

Suite ID format: cs-{kem}-{aead}-{sig}
  Example: cs-mlkem768-aesgcm-mldsa65

4. KEM Keygen Performance (Live Benchmark Data)



5. KEM Primitive Times – Measured Data (Baseline)

Algorithm	Keygen (ms)	Encaps (ms)	Decaps (ms)	PK Size (B)
ML-KEM-512	0.25	0.27	0.14	800
ML-KEM-768	0.30	0.32	0.13	1,184
ML-KEM-1024	0.20	0.30	0.16	1,568
HQC-128	2.77	44.90	5.42	2,249
HQC-192	4.67	136.41	15.47	4,522
HQC-256	8.00	249.81	25.27	7,245
McEliece-348864	134.08	0.72	18.71	261,120
McEliece-460896	180.21	1.40	57.66	524,160
McEliece-8192128	378.10	3.17	139.81	1,357,824

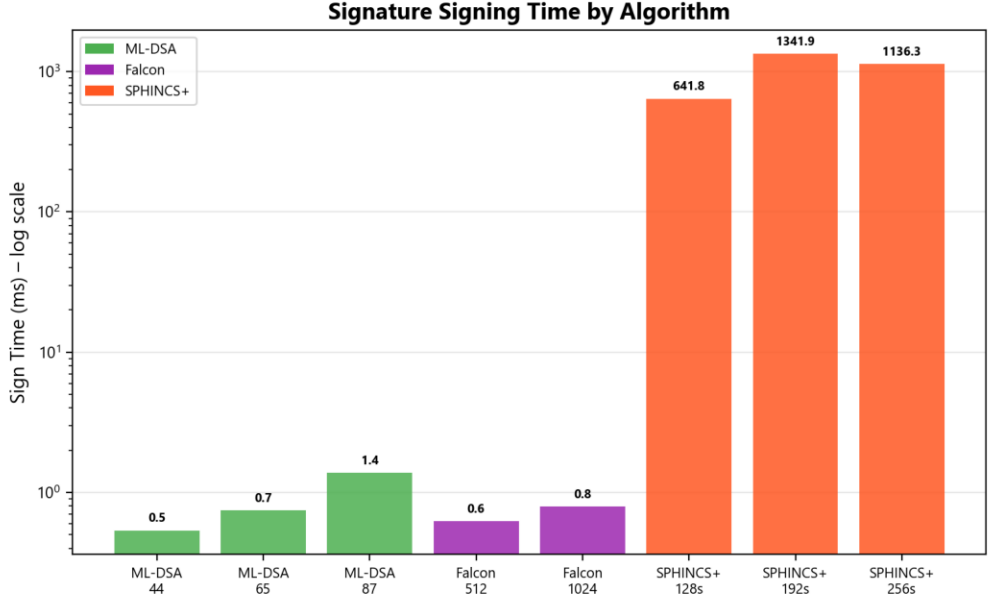
Note: These are medians from live end-to-end benchmark runs (not isolated primitive benchmarks).
Values include real system load, GC pauses, and scheduling noise.

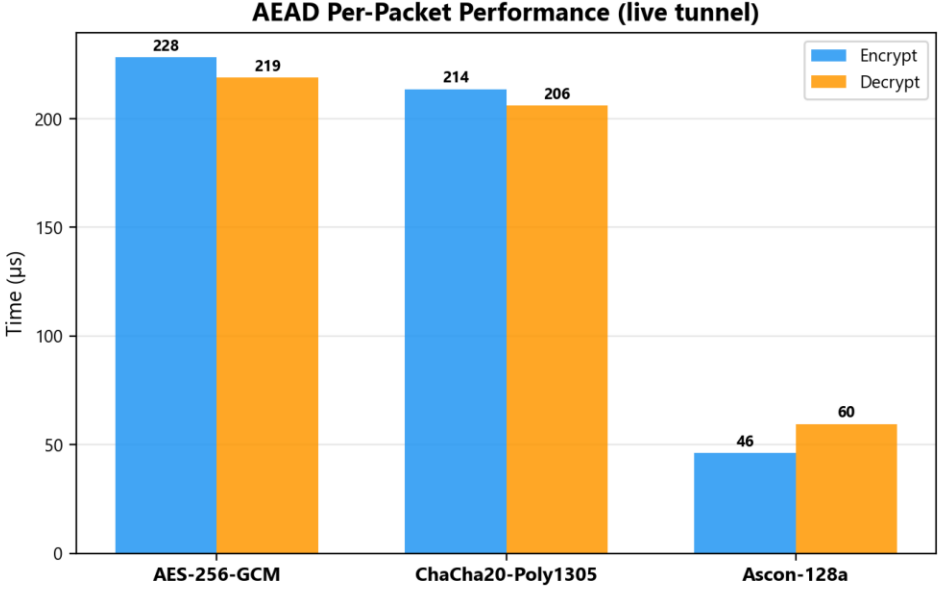
6. Signature Primitive Times – Measured Data (Baseline)

Algorithm	Sign (ms)	Verify (ms)	Sig Size (B)
ML-DSA-44	0.54	0.76	2,420
ML-DSA-65	0.74	0.88	3,309
ML-DSA-87	1.38	1.17	4,627
Falcon-512	0.62	0.60	654
Falcon-1024	0.79	0.71	1,271
SPHINCS+-128s	641.83	1.98	7,856
SPHINCS+-192s	1341.94	2.80	16,224
SPHINCS+-256s	1136.29	3.83	29,792

SPHINCS+ signing dominates the handshake for any suite that includes it.
Note: SIG keygen is performed offline (pre-distributed keys); only sign + verify affect handshake time.

7. Signature & AEAD Performance Charts





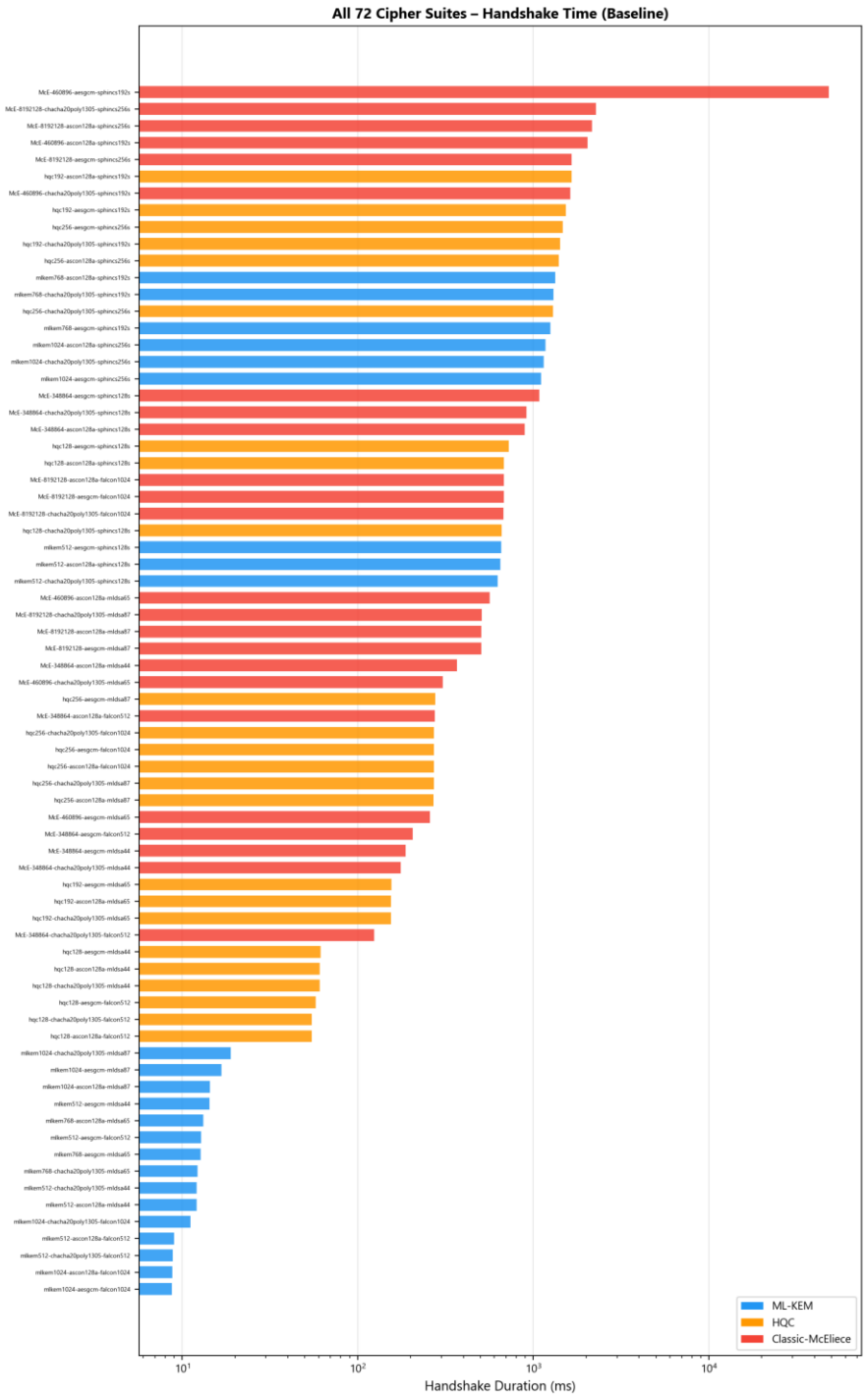
8. AEAD Per-Packet Performance (Live Tunnel)

Algorithm	Encrypt	Decrypt	Relative to Ascon
AES-256-GCM	228.3 µs	219.0 µs	4.95×
ChaCha20-Poly1305	213.6 µs	206.2 µs	4.63×
Ascon-128a	46.1 µs	59.5 µs	1.00×

Note: These are per-packet averages from live tunnel operation (not isolated microbenchmarks).
They include full AEAD framing overhead: header construction, AAD binding, nonce reconstruction.

At 320 Hz MAVLink rate, even the slowest AEAD adds only ~73 µs/s of CPU time = 0.007% overhead.
Conclusion: AEAD algorithm choice has NO meaningful impact on system performance.

9. End-to-End Handshake: All 72 Suites (Baseline)



10. Handshake Statistics by NIST Level & KEM Family

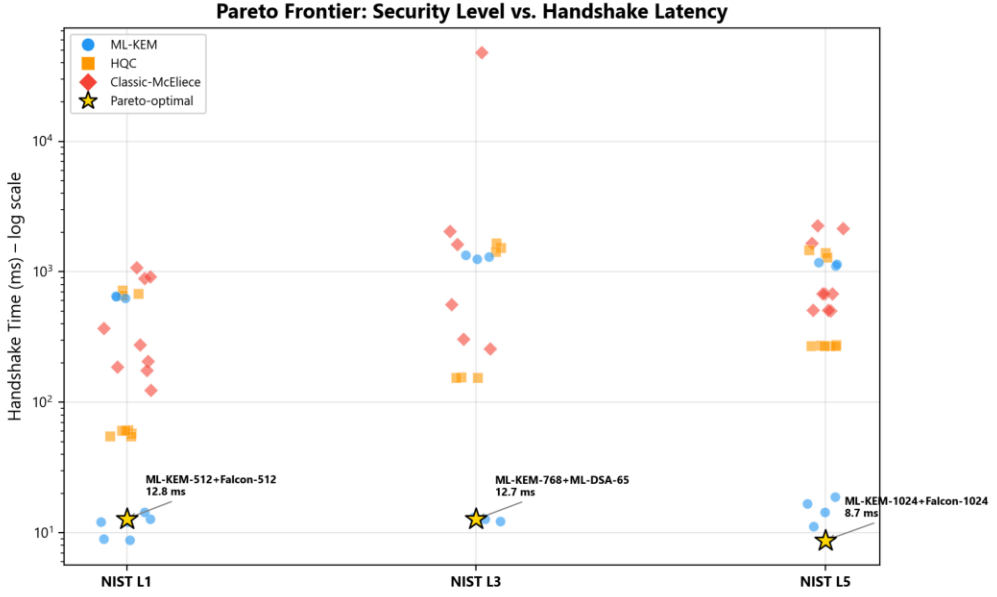
Level	n	Mean (ms)	Median (ms)	P95 (ms)	Max (ms)
L1	27	320.1	176.2	910.2	1079.9
L3	18	3443.7	906.6	8962.5	48185.7
L5	27	701.6	507.1	2008.3	2269.1

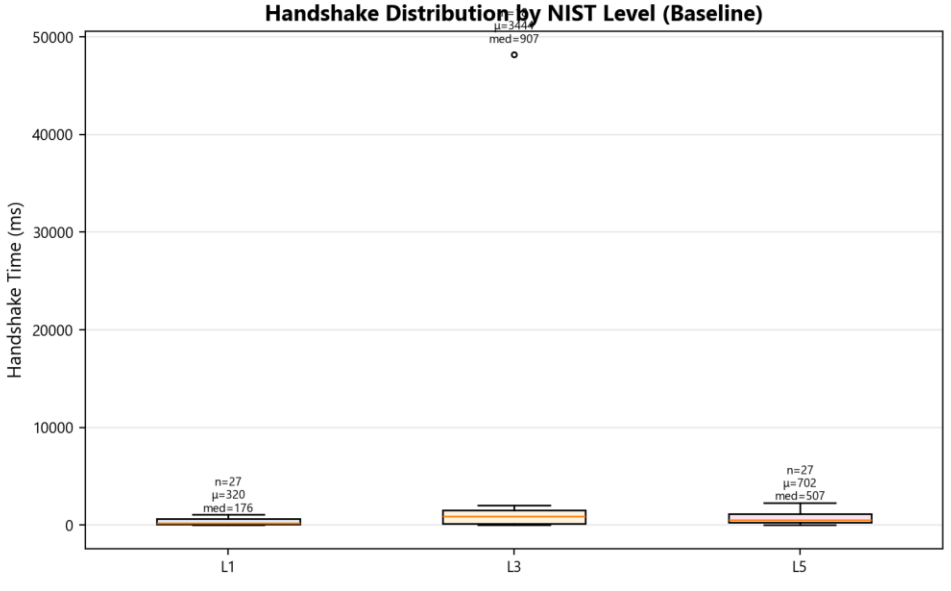
By NIST Security Level ↑ **By KEM Family ↓**

KEM Family	n	Mean (ms)	Median (ms)	P95 (ms)	Max (ms)
ML-KEM	24	393.4	14.4	1297.3	1337.5
HQC	24	553.8	271.4	1526.9	1650.8
Classic-McEliece	24	2785.0	620.1	2253.0	48185.7

71/72 suites succeeded in baseline. 1 timeout: likely McEliece-460896 + SPHINCS+-192s.
L3 shows high max (48,186 ms) due to this outlier timeout value.

11. Pareto Frontier & Handshake Distribution





12. Pareto-Optimal Suites (Measured Data)

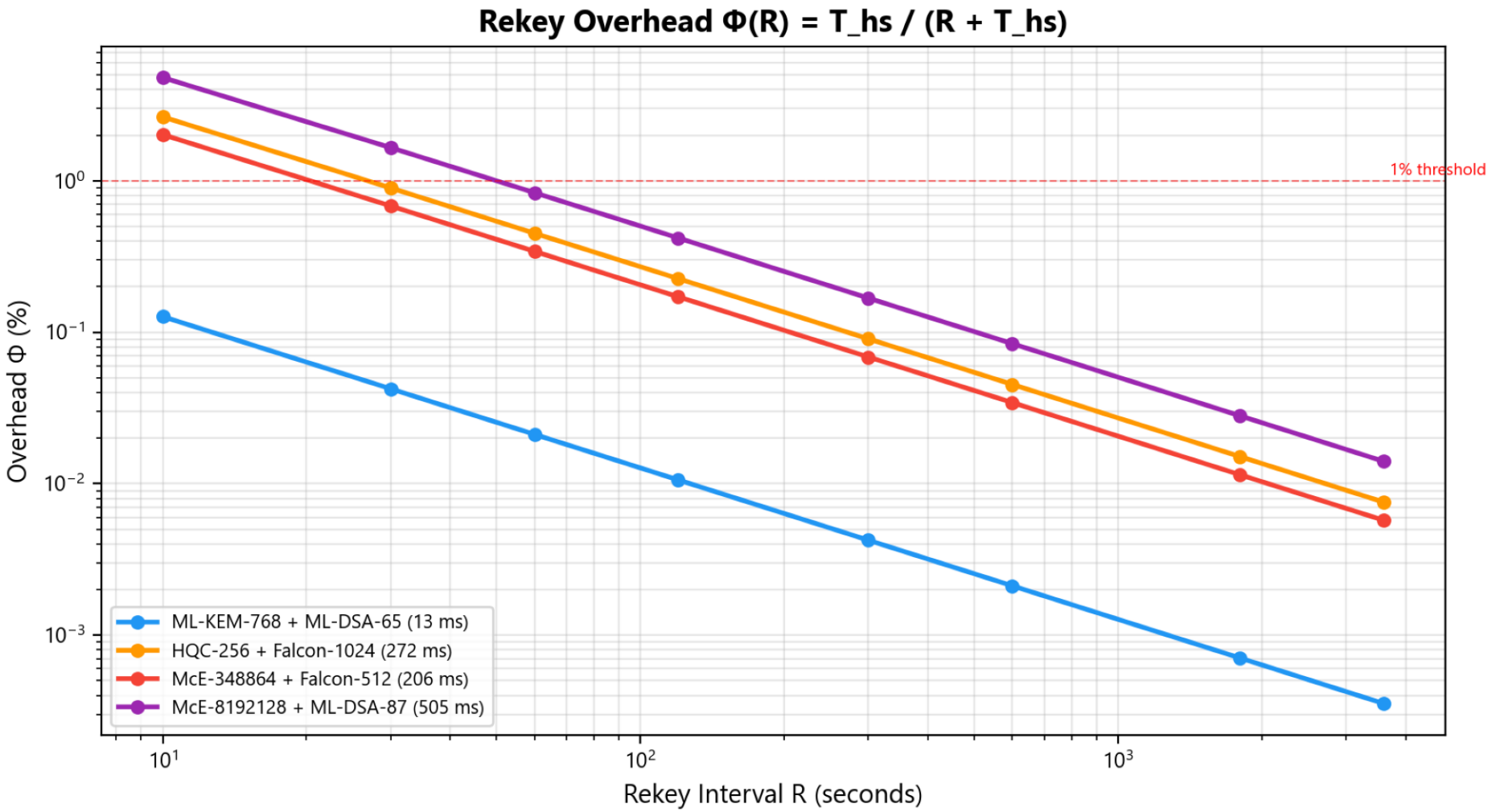
Suite (KEM + SIG)	NIST Level	T_hs (ms)	PK Size (B)	Φ(R=60s)
ML-KEM-512 + Falcon-512	L1	12.80	800	0.0213%
ML-KEM-768 + ML-DSA-65	L3	12.71	1,184	0.0212%
ML-KEM-1024 + Falcon-1024	L5	8.75	1,568	0.0146%

Rekey Overhead Formula: $\Phi(R) = T_{hs} / (R + T_{hs})$

At R = 60s rekey interval, all three Pareto-optimal suites have Φ < 0.03%.
The overhead is negligible — the tunnel can rekey every minute with zero impact.

All Pareto-optimal suites use ML-KEM. No HQC or McEliece suite
appears on the frontier because their KEM operations dominate handshake time.

13. Rekey Overhead Φ(R) Analysis

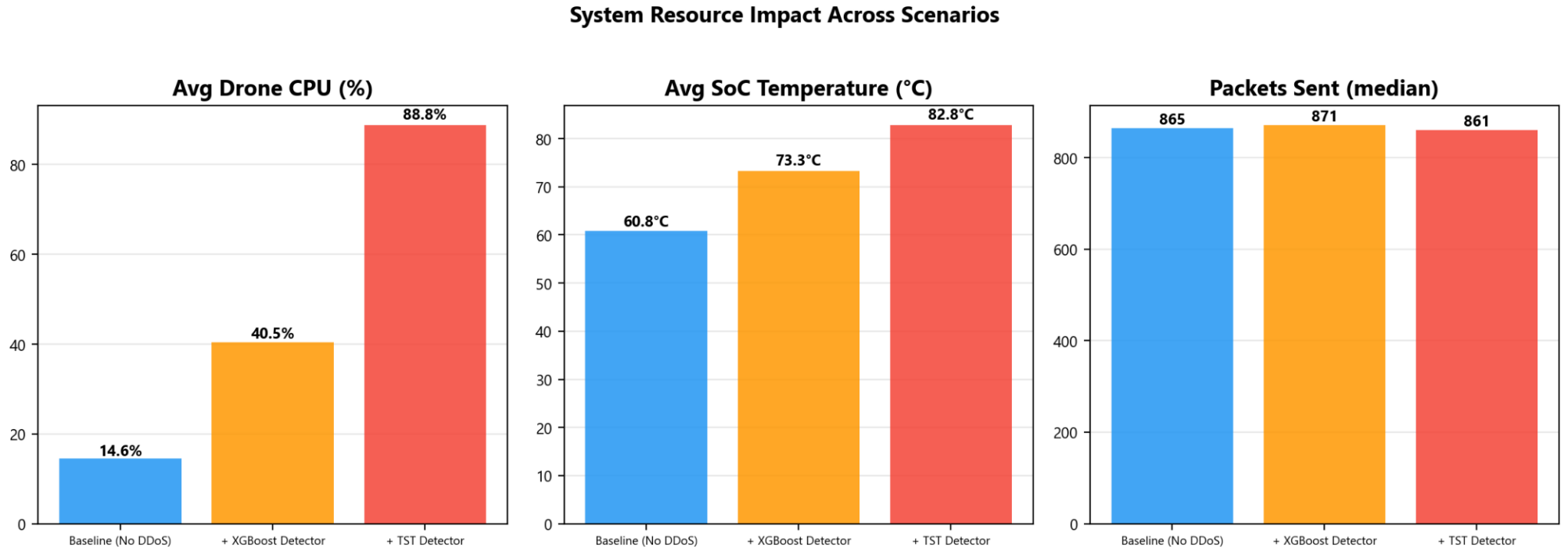


14. DDoS Detection Models

Property	XGBoost Detector	TST Detector
Architecture	XGBoost (Gradient Boosted Trees)	Time Series Transformer (3-layer, 16-head, d=128)
Window Size	5 packets	400 packets
Detection Latency	~3 seconds	~240 seconds
Inference Time	~microseconds (µs)	~milliseconds (ms)
Threading Model	Single-thread, GIL-friendly	Single-thread, CPU-only
Model Size	~100 KB	~5 MB
Feature Extraction	Packet count per 0.6s window	Packet count per 0.6s window
Output	Binary (Attack/Normal)	Binary (Attack/Normal)
Best For	Real-time flight detection	Post-flight analysis

Detection mechanism: Scapy sniffs wlan0 for MAVLink v2 (0xFD magic byte).
Normal traffic: ~32 packets/window. DDoS attack: ~5–14 packets/window.
Hybrid cascaded pipeline: XGBoost (fast screening) → TST (deep confirmation).

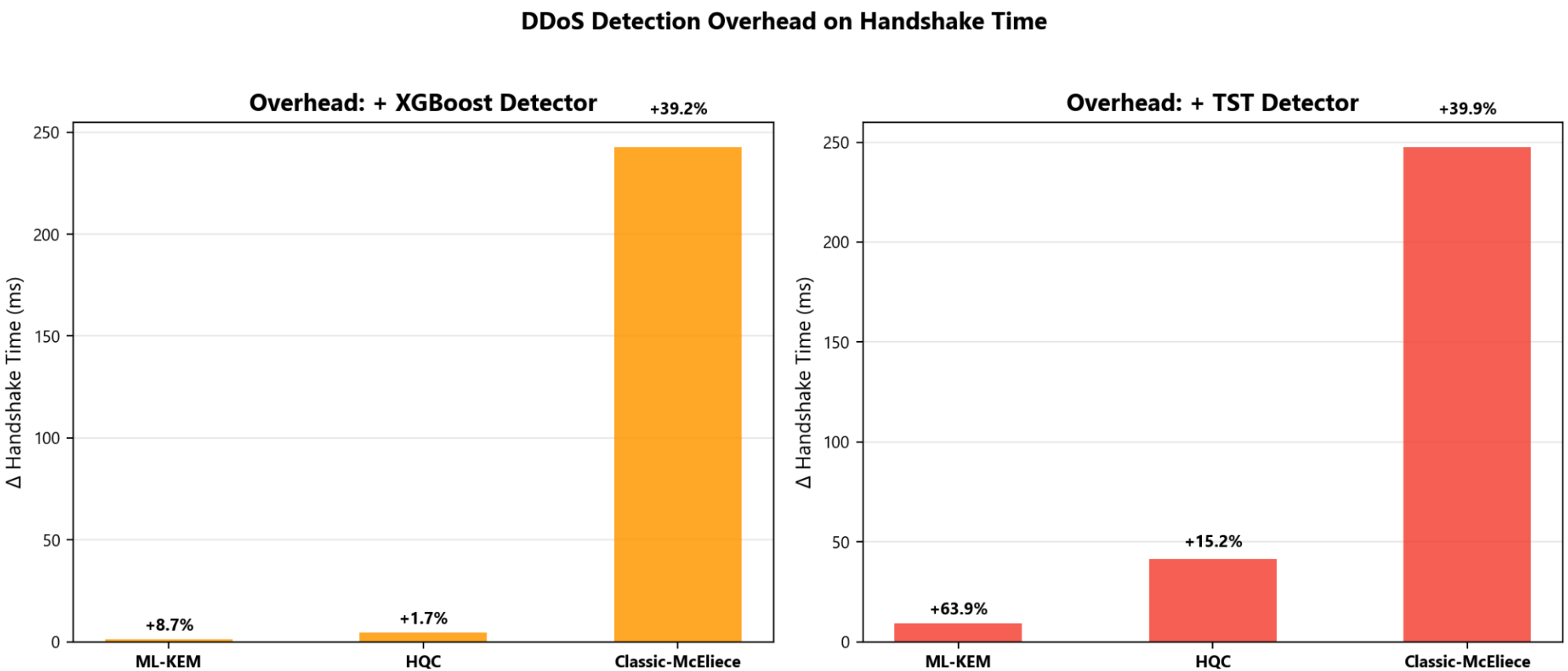
15. Cross-Scenario System Metrics



16. Cross-Scenario Comparison – Full Data Table

Metric	Baseline (No DDoS)	+ XGBoost Detector	+ TST Detector
Median Handshake (ms)	290.5	277.9	350.1
Mean Handshake (ms)	1244.1	1258.5	1386.4
P95 Handshake (ms)	1826.3	1737.0	2275.5
Avg CPU (%)	14.6	40.5	88.8
Peak CPU (%)	38.3	62.8	95.4
Avg Temp (°C)	60.8	73.3	82.8
Packets Sent (med)	865	871	861
Packet Loss (%)	0.00	0.00	0.00
Suites Passed	71/72	71/72	71/72

17. DDoS Overhead Analysis



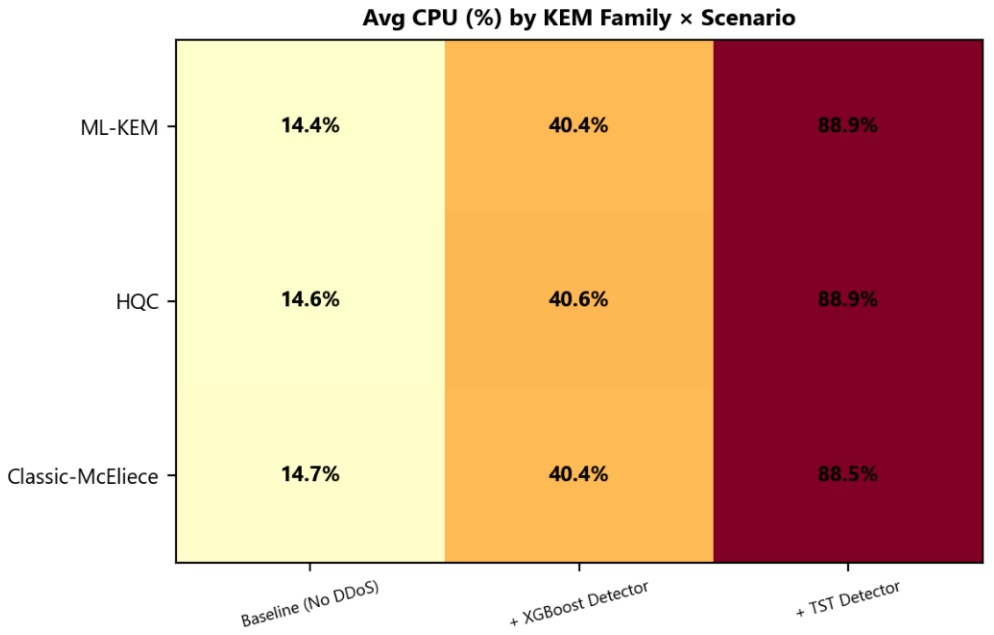
18. DDoS Overhead – Detailed Numbers

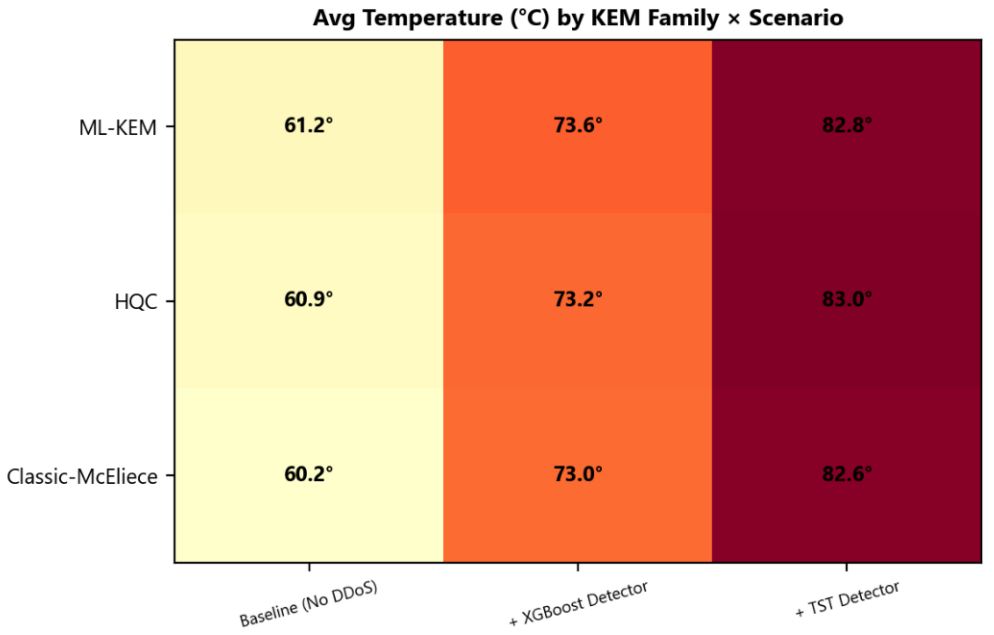
KEM Family	Scenario	Baseline (ms)	With Detector (ms)	Δ (ms)	Δ (%)
ML-KEM	+ XGBoost	14.4	15.6	+1.3	+8.7%
ML-KEM	+ TST	14.4	23.5	+9.2	+63.9%
HQC	+ XGBoost	271.4	275.9	+4.5	+1.7%
HQC	+ TST	271.4	312.7	+41.3	+15.2%
Classic-McEliece	+ XGBoost	620.1	862.9	+242.8	+39.2%
Classic-McEliece	+ TST	620.1	867.8	+247.6	+39.9%

Analysis:

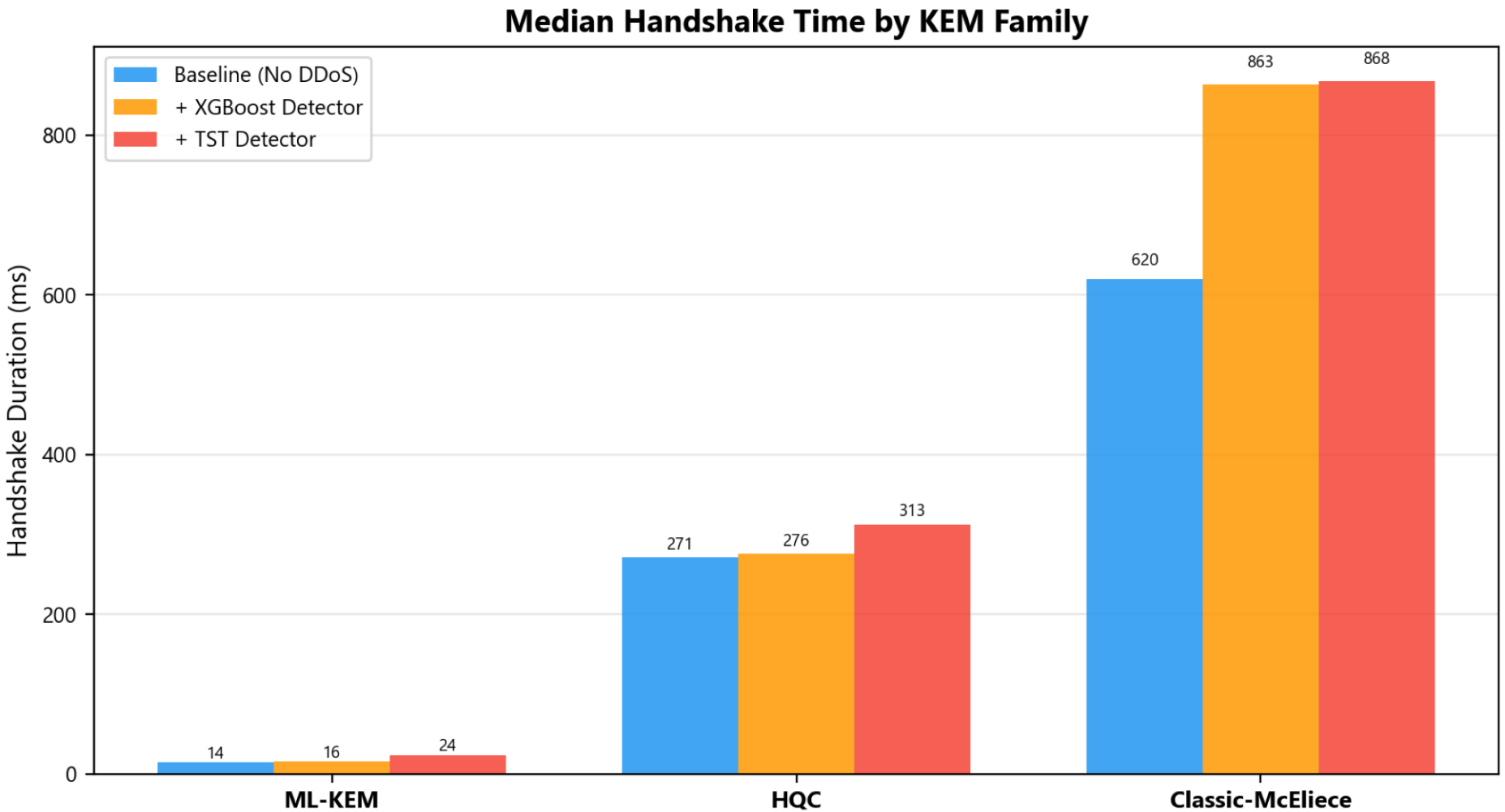
- XGBoost overhead on ML-KEM is minimal (~+1 ms, ~+9%) — recommend for flight
- TST overhead is higher (~+9 ms on ML-KEM, +64%) due to Transformer weight in memory
- McEliece shows large absolute Δ in both scenarios due to keygen sensitivity to CPU load
- TST raises avg CPU from 14.6% → 88.8% and temp from 60.8°C → 82.8°C (near throttle!)
- XGBoost raises avg CPU to 40.5% and temp to 73.3°C — manageable

19. CPU & Temperature Heatmap (KEM Family × Scenario)





20. Handshake by KEM Family Across Scenarios



21. Key Findings & Conclusions

```
Finding 1: Algorithm-FAMILY selection dominates performance
  ML-KEM median handshake: 14.4 ms  vs  McEliece: 620.1 ms  (43× slower)
  Switching McEliece→ML-KEM at same NIST level = 43× speedup, ZERO security loss

Finding 2: SPHINCS+ signing is the biggest single bottleneck
  SPHINCS+-128s: 642 ms sign  |  SPHINCS+-192s: 1,342 ms sign
  Any suite with SPHINCS+ is dominated by the signature, not the KEM

Finding 3: AEAD choice is a complete non-factor
  Max difference: ~182 µs/packet (AES vs Ascon)
  At 320 Hz: ~58 ms/s additional = 0.006% overhead — invisible

Finding 4: XGBoost detection has minimal overhead
  ML-KEM handshake: +1.3 ms (+8.7%), CPU: +26%, temp: +12.5°C
  Viable for in-flight detection

Finding 5: TST detection causes near-throttle conditions
  CPU: 14.6% → 88.8%, temp: 60.8°C → 82.8°C (throttle at 80°C!)
  ML-KEM handshake: +9.2 ms (+64%)
  Not recommended for flight — use for post-flight analysis

Finding 6: All three Pareto-optimal suites use ML-KEM
  ★ L1: ML-KEM-512 + Falcon-512     = 12.8 ms
  ★ L3: ML-KEM-768 + ML-DSA-65      = 12.7 ms
  ★ L5: ML-KEM-1024 + Falcon-1024   = 8.7 ms
  Rekey overhead at R=60s: < 0.022% for all three

Recommended default suite: ML-KEM-768 + ML-DSA-65 + AES-256-GCM (NIST L3)
Recommended detector: XGBoost for flight, TST for ground analysis
```