Testing ML-KEM Implementations for Side-channel Vulnerability: The Airtight Framework

Isaiah Seals, Douglas Hodson and Mark Reith
Air Force Institute of Technology, Dayton, USA

isaiah.seals.1@au.af.edu
douglas.hodson@au.af.edu
mark.reith.3@au.af.edu

Abstract: As quantum computing advances toward operational capability, traditional public-key communication schemes are becoming obsolete. Algorithms like Rivest-Shamir-Adleman and Elliptic Curve Cryptography cryptosystems, foundational to current secure communications, are vulnerable to Shor's algorithm, which can efficiently factor and compute discrete logarithms once large-scale quantum systems emerge. In anticipation of this threat, the National Institute of Standards and Technology (NIST) has standardized the Module-Lattice Key Encapsulation Mechanism (ML-KEM), derived from the Crystals KYBER family, as a post-quantum cryptographic (PQC) solution designed to secure communications against both classical and quantum attackers. However, while ML-KEM's lattice-based cryptographic system provides provable resistance to cryptanalytic attacks, its real-world implementations are susceptible to side-channel attacks (SCA). These attacks, which exploit timing variations, power consumption, or electromagnetic emissions, bypass the algorithm's theoretical security by extracting secret keys from physical leakage. Even as cryptographic design moves beyond quantum threats, physical-layer vulnerabilities reintroduce risk at the hardware level. These physical-layer security vulnerabilities are a looming threat to future PQC security. Given the dangers posed by correlation power analysis, cryptographic programs must be implemented to mitigate these vulnerabilities, or else the increased security provided by PQC methods will be in vain. In this paper, we assert that the secure adoption of PQC in defense communication systems demands accelerated implementation of ML-KEM within secure communication protocols to ensure post-quantum readiness and rigorous evaluation of its side-channel resilience through experimental validation and countermeasure integration. We survey recent research into ML-KEM side-channel resistance, identify existing countermeasure frameworks (masking, constant-time operations, noise injection), and propose Airtight: a framework for standardized testing of secure communication methods.

Keywords: Post-quantum cryptography, Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM), Side-Channel Attacks (SCA), Power analysis, Secure implementations, Defense communications

1. Introduction

Quantum computing poses an evolving and rapidly growing threat to current cryptographic methods. A quantum computer implementing Shor's algorithm can break RSA, one of the most commonly deployed cryptographic schemes, once a high enough number of logical qubits is achieved (Nehal, Farhan and Sunad, 2020). In response to the growing capabilities of quantum computers, NIST standardized ML-KEM as a post-quantum key encapsulation mechanism (KEM).

As current systems grow increasingly vulnerable to quantum attacks, the transition to post-quantum cryptography must be prioritized. However, implementing ML-KEM poses significant challenges, particularly in ensuring its security against side-channel attacks. These attacks exploit information leaked through various channels, such as power consumption, timing, and electromagnetic emissions.

Some organizations have attempted to delay quantum risk by transitioning to larger RSA key sizes (U.S. Department of Defense, 2023), but this is a temporary solution to a longer-term problem. A new scheme, one secure against quantum computing, is required. ML-KEM, as a standardized post-quantum KEM, is a promising candidate. Nevertheless, its security relies heavily on proper implementation, which can be compromised by side-channel attacks.

Before ML-KEM can be widely adopted, it must be tested thoroughly against side-channel attacks. This paper partially addresses the need for comprehensive testing by proposing a testing framework, called Airtight, which focuses on identifying and mitigating side-channel vulnerabilities in ML-KEM implementations.

The rest of this paper is organized as follows: Sections 2 and 3 discuss the ML-KEM scheme and attacks on the communication system, specifically in relation to side-channel attacks. Section 4 explores common countermeasures against side-channel attacks and how attackers may try to circumvent these defensive strategies. Finally, Section 5 presents Airtight, a general framework for testing the side-channel attack security of ML-KEM for implementation in secure systems.

2. The ML-KEM Scheme and Security

ML-KEM relies on lattice-based cryptography to provide key encapsulation for asymmetric communication. NIST standardized ML-KEM in 2024 because of its theoretical security against quantum computing (NIST, 2024).

ML-KEM derives its computational hardness from the Module Learning with Errors (MLWE) problem (Marmebro and Stenbom, 2024; NIST, 2024). MLWE is a generalization of the Learning with Errors Problem (LWE). In LWE, attackers analyze a set of "noisy" linear equations after perturbation through introduction of error values (NIST, 2024). This perturbation makes solving for the set of linear equations computationally infeasible for attackers.

In the context of lattice-based cryptography, LWE is computationally difficult because of a set of lattice problems, including the Shortest Independent Vectors Problem (SIVP) and Closest Vector Problem (CVP). These problems relate to determining the closest vector on a lattice to a point in space, and depend on the basis vectors used to traverse the lattice. When using orthogonal bases, lattice traversal, and by extension, these problems, are comparatively easy. When bad bases are used, finding solutions to these problems within a lattice becomes computationally infeasible. Importantly, Regev (2009) proved that an average-case implementation of a public-key cryptosystem using these problems for its security can be reduced to a worst-case problem for attackers, keeping the system more secure.

The ML-KEM scheme, like all KEM schemes, includes three main steps: KeyGen, Encaps, and Decaps. The following is a short discussion of the functions:

2.1 ML-KEM.KeyGen

In KeyGen, randomness is generated internally to create the encapsulation and decapsulation keys. The encapsulation key, commonly known as the public key, is created from a set of random noisy linear equations in the Number Theoretic Transform domain:

$\hat{t} \leftarrow \hat{A} \circ \hat{s} + \hat{e}$

The decapsulation key is generated from the same randomness used above, and is the byte-encoded $\hat{\mathbf{s}}$ vector.

2.2 ML-KEM.Encaps

Encaps uses internally-generated randomness to generate a message *m*, then encrypts *m*, outputting (*K*, *c*). The encryption function also uses the mathematical security of LWE, generating a vector and a value from the set of noisy linear equations which are concatenated and sent to the receiver as the ciphertext.

$u \leftarrow NTT^{-1}(\hat{A}^{T} \circ \hat{y}) + e_1$

$v \leftarrow NTT^{-1}(\hat{t}^{T} \circ \hat{y}) + e_2 + \mu$

The shared secret key *K* and ciphertext *c* are then sent for decapsulation.

2.3 ML-KEM.Decaps

The Decaps function uses the decapsulation key to decrypt the ciphertext, recovering the message *m′*.

$w \leftarrow v' - NTT^{-1}(\hat{s}^{T} \circ NTT(u'))$

After message recovery, a hash function is used to generate a shared secret key *K′*. Decaps also re-uses the encryption function to calculate *c′* using *m′* to compare with the given ciphertext. If *c=c′*, then *K′* is the correct shared secret key and symmetric encrypted communication can begin.

Notably, decryption occurs whether or not $c \neq c'$, which exposes ML-KEM to possible side-channel attacks at the decryption step. Next, this paper discusses side-channel attacks on ML-KEM, including an in-depth discussion of Correlation Power Analysis (CPA) attacks.

3. Side-channel Attacks on ML-KEM

Although ML-KEM resists quantum cryptanalysis, attackers can still successfully exploit side-channel leakage in its implementations. Specifically, CPA attacks on implementations of ML-KEM pose a looming threat (Primas, Pessl and Mangard, 2017; Kennaway et al., 2025). Side-channel attacks introduce multiple attack vectors for consideration when creating cryptosystems, as seen in Fig. 1.

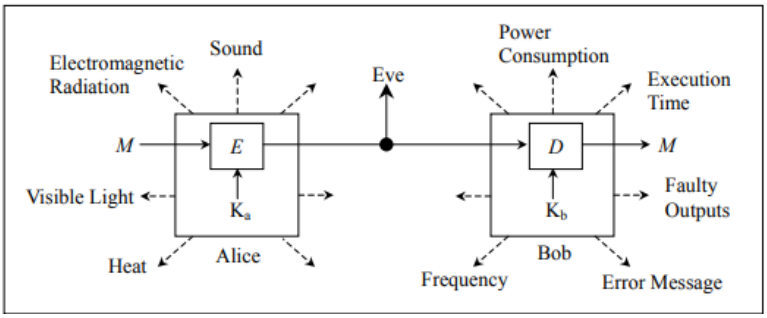

Figure 1: Attack vectors added by side-channel attacks (Zhou and Feng, 2005)

3.1 Timing Attacks

Timing attacks involve measuring the amount of time taken for a function to complete. The differences in expected versus actual function execution times give attackers information about the system's internal state. Timing attacks may be defended against comparatively easily by introducing constant-time algorithms (Arriaga, Barbosa and Jarecki, 2025).

3.2 Power Analysis Attacks

The two power analysis attacks discussed here are correlation power analysis (CPA) attacks and differential power analysis (DPA) attacks. Both attempt to discover secret information by determining points of interest in schema execution, then analyzing the power differential after a function is completed on data. Attackers usually determine sensitive information by calculating the Hamming weight and Hamming distance of intermediate values. Fig. 2 shows trace values during point of interest discovery for a power analysis attack.

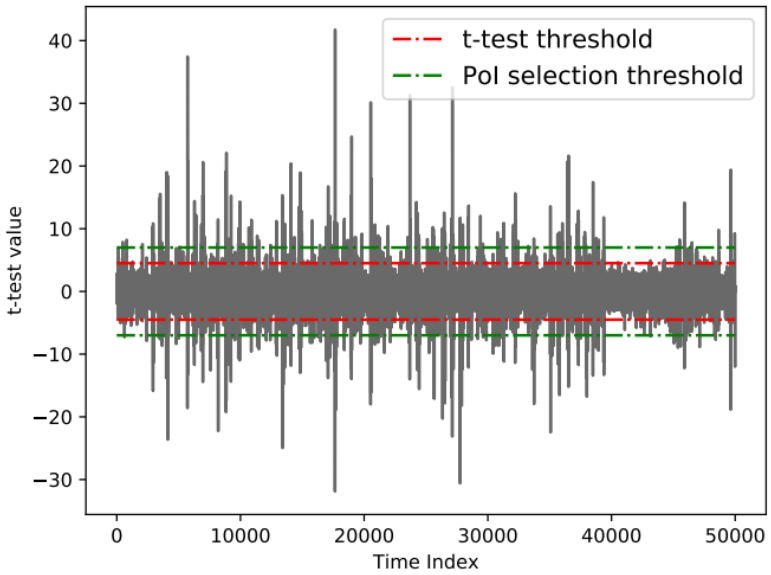

Figure 2: Point of interest discovery for power analysis attack on KYBER (Ravi et al., 2020)

3.2.1 Differential power analysis attacks

Differential power analysis (DPA) attacks typically use the difference of mean calculation to determine sensitive information in power consumption during task execution.

3.2.2 Correlation power analysis attacks

CPA attacks are another subset of power analysis attacks and typically use the Pearson correlation coefficient to calculate the correlation between power difference and sensitive data.

Side-channel attacks on cryptosystems are greatly strengthened by profiling the communication system used. A profiled attack occurs when an attacker has an identical copy of the communication system being used, creating a larger attack surface. The vast majority of non-profiling attacks on ML-KEM target the polynomial multiplication step, but in profiled attacks, many different functions are targeted (Kennaway et al., 2025).

When designing a security testing framework, researchers must assume that attackers can mount profiled side-channel attacks.

In addition to profiling capabilities, deep-learning has enhanced SCA capabilities due to its usefulness in cluster recognition and pattern analysis (Kennaway et al., 2025). Because of the possible increased use of deep-learning in future side-channel attacks, a framework for securing cryptosystems against these attacks must address the security vulnerability created by deep-learning power analysis attacks.

Although ML-KEM is secure enough to use in PQC communications, it, along with other lattice-based cryptosystems, remains vulnerable to side-channel attacks (Primas, Pessl and Mangard, 2017). Some countermeasures have proven effective against these threats, but the increased sophistication of possible attacks due to deep-learning, along with research in single-trace power analysis attacks, continues to pose a threat. In the next section, this paper discusses countermeasures against side-channel attacks, specifically for ML-KEM and other lattice-based cryptosystems.

4. Countermeasures Against Side-channel Attacks

Although side-channel attacks (SCAs) expose vulnerabilities at the physical and implementation layers, a variety of countermeasures have been developed to mitigate these risks. For ML-KEM and other lattice-based cryptographic systems, these defenses attempt to ensure that operations remain secure even under direct power, timing, or electromagnetic analysis (Eldib and Wang, 2014). Effective countermeasure designs must protect against key leakage without imposing excessive computational or hardware overhead. This section reviews the common types of SCA countermeasures, evaluates their effectiveness as described in the recent literature, and highlights the continuing challenges in standardizing protections for post-quantum cryptographic implementations.

Table 1 clarifies the strengths and weaknesses of different side-channel attack countermeasures to be discussed in the following paragraphs. These trade-offs must be considered when developing a framework for testing ML-KEM and other lattice-based KEM implementations in the future to reduce the attack surface on the new communication scheme.

Table 1: Countermeasure categories for ML-KEM implementations

Type	Strength	Weakness
Constant-Time Execution	Strong against timing side channels	Still leaks power/EM traces
Masking	Protects against first-order CPA	Breakable by higher-order attacks; high overhead in NTT ops
Noise Injection	Resists basic DPA and CPA	Machine learning overcomes noise
Hardware Defenses	Best long-term physical security	High cost; not practical for legacy systems

4.1 Countermeasures Against Timing Attacks

Developers counter timing attacks by designing functions that execute in constant time regardless of secret data (Arriaga, Barbosa and Jarecki, 2025). This can be implemented in multiple ways. One common method is to rework encoding and transform functions with rejection sampling to maintain the uses of core functions while eliminating time variation due to rejections (Eldib and Wang, 2014). This can be accomplished by encoding public keys as byte strings. A trade-off of this method is larger public key sizes, leading to slower computation times.

4.2 Countermeasures Against Power Analysis Attacks

The two major countermeasures against power analysis attacks are masking and noise injection. Masking techniques statistically decouple intermediate computations from secret key values, leading to perfect masking of one function (Eldib and Wang, 2014). This defeats first-order CPA and DPA attacks; however, higher-order CPA and DPA techniques can still defeat perfectly masked implementations, especially when profiling or deep-learning assistance is available (Ravi et al., 2020; Kennaway et al., 2025). Masking also imposes significant computational and memory overhead, as each masked operation requires additional randomness and duplicated computation paths. These resource demands make masking particularly difficult to deploy in lightweight or embedded environments where power, latency, and hardware area are tightly constrained (Ravi et al., 2024b). Noise injection is another technique for countering CPA and DPA attacks.

Noise injection makes finding points of interest difficult by introducing randomness into power emittance during function execution. Nonetheless, noise injection can also be defeated with deep-learning classifiers capable of filtering or reconstructing signal leakage.

4.3 Common Methods for detecting Chosen-ciphertext Attacks

Two common methods exist for detecting and countering chosen-ciphertext attacks, as described in Ravi et al. (2024a). The first involves checking ciphertext values and ignoring any ciphertext sent for decapsulation if it does not meet an entropy threshold. This check allows for pre-decryption rejection of suspicious ciphertexts, but in practice, the method remains vulnerable to masked chosen-ciphertext attacks because adversaries can craft ciphertexts that pass entropy validation while still leaking key material. The second method detects failure during decapsulation, typically resulting from a chosen-ciphertext attack (Ravi et al., 2024a). ML-KEM implements this approach by re-encrypting the decrypted message and ensuring $c = c'$ (NIST, 2024). This failure check remains widely adopted because most chosen-ciphertext attacks require multiple malicious ciphertexts before a key can be recovered. Nonetheless, attackers can still defeat this defense by exploiting valid ciphertexts and leveraging statistical leakage, as demonstrated in Ravi et al. (2024a).

4.4 Secure Hardware Implementations

Implementing cryptosystems on secure hardware can also help prevent attacks, specifically CPA and DPA attacks (Ravi et al., 2024b). For example, using shielded hardware to emit less power during execution or using algorithmically noisy devices that have a low signal-to-noise ratio can also deter attacks (Ravi et al., 2024b). Secure hardware implementations can be effective countermeasures to side-channel attacks because they offer longer-term physical security. Implementations of ML-KEM must use secure hardware from the first implementation because transitioning from legacy systems to secure hardware implementations takes time and is expensive. Additionally, with increasing adoption of cloud and virtualized computing environments, security professionals must ensure that real-time power consumption and performance telemetry are not exposed through cloud monitoring channels, as remote leakage of power traces may enable CPA or DPA attacks without requiring physical device access.

4.5 Black-box Implementations

Black-box implementations of cryptosystems are often considered more secure because an attacker may only analyze input and output. However, SCA attacks mean that black-box implementations of any cryptosystem are practically impossible to achieve. These schemes can be hardened to include as many countermeasures to side-channel attacks as possible, but Ravi et al. (2020), Ravi et al. (2024a), and Kennaway et al. (2025) show that almost any countermeasure against SCA attacks can be defeated. Attacks that defeat these countermeasures must be considered when developing a testing framework for PQC cryptosystems.

5. The Airtight Framework

Airtight is a proposed testing and validation structure designed to evaluate the side-channel resilience of ML-KEM implementations before they are fielded in secure communication systems. While the cryptographic design of ML-KEM provides worst-case to average-case hardness guarantees against quantum and classical adversaries (Regev, 2009; NIST, 2024), its real-world implementations may still leak sensitive information through timing, power, or electromagnetic emissions (Primas, Pessl and Mangard, 2017; Kennaway et al., 2025). The Airtight framework bridges the gap between theoretical security and real-world assurance by providing standardized testing, evaluation, and integration strategies for ML-KEM within defense communication systems.

5.1 Framework Overview

Airtight consists of three modular phases aligned with DoD cryptographic agility goals and the PQC transition requirements for future secure networks (U.S. Department of Defense, 2023):

- Leakage Detection in Controlled Environments
- Validated Countermeasure Integration
- Operational Deployment and Certification

Each phase produces quantifiable metrics that evaluators use to certify whether an ML-KEM implementation meets mission security requirements.

5.2 Phase 1: Leakage Detection and Profiling

The first phase focuses on establishing a controlled test environment equipped with power, timing, and electromagnetic measurement instruments. Implementations are profiled using industry-standard techniques such as Test Vector Leakage Assessment (TVLA) and correlation power analysis (CPA), alongside timing analysis methods designed to detect variable-latency execution. The framework defines standard parameters such as:

- minimum of 50k–100k trace captures for TVLA evaluation (Unger et al., 2022)
- t-value thresholds of $|t| > 4.5$ to indicate statistically significant leakage
- targeting NTT multiplication, rejection sampling, and decapsulation comparison logic as primary points of interest (Kennaway et al., 2025)
- timing jitter and execution-path variance measurements to detect deviation from constant-time behaviour

Profiling in Phase I identifies vulnerable execution windows before any defensive measures are applied, ensuring that countermeasures are evaluated from a data-driven baseline and that both power-based and timing-based leakage are characterized early in the evaluation process.

5.3 Phase 2: Countermeasure Testing and Metrics

The second phase integrates SCA countermeasures such as masking, constant-time arithmetic, and noise injection. Airtight evaluates both the reduction of side-channel leakage and the operational cost of the countermeasure. This dual assessment allows security effectiveness to be weighed against performance and resource demands, including the mitigation of timing leakage:

- Leakage Reduction: percentage decrease in correlation or variance relative to unprotected traces.
- Performance Overhead: increase in cycle count, power draw, or memory size
- Break Resistance: resistance to profiling-based CPA, higher-order DPA, and deep-learning–enhanced SCA (Kennaway et al., 2025). Scored 0-3, dependent on if the system is resistant to first-order, higher-order, and deep-learning enhanced CPA and DPA attacks, with the score incrementing for each type of attack the implementation is resistant against.
- Timing Leakage Mitigation: reduction in execution-path variance, jitter, and branch-dependent latency relative to baseline; verification of constant-time implementations through repeated statistical timing analysis

This scoring system enables consistent comparison across hardware platforms, microarchitectures, or compiler configurations while capturing both power/EM and timing-based leakage characteristics.

5.4 Phase 3: Operational Deployment and Certification

The last phase aligns tested implementations with ML-KEM deployment guidelines from FIPS 203/204 (Marmebro and Stenbom, 2024; NIST, 2024). Implementations that pass Phase II testing are packaged with mandatory metadata including leakage profiles, required hardware constraints, and recommended compiler settings. Airtight proposes that DoD-approved implementations be given a Side-Channel Security Certification Level (SCSL) score derived from Phase II metrics, enabling predictable reuse and procurement contracting.

Finally, because the framework treats leakage detection and countermeasure scoring as algorithm-agnostic, Airtight can be reused to test future PQC schemes such as ML-DSA or BIKE, enabling a larger DoD cryptographic assurance pipeline (U.S. Department of Defense, 2023). This modularity enables repeatable validation and reduces algorithm-transition costs over the next decade of PQC standardization.

Because of Airtight's modularity and capability to be used in myriad organizations, no single scoring system can be proposed. Each organization should develop its own method of validating security with a scoring system tailored to its needs. For example, an organization that values confidentiality may give less weight to performance overhead penalties, while an organization that requires lightweight communications systems may value that metric more. An opportunity for future work in this area would be to further develop scoring frameworks for stereotypical organizations in different fields, using data-driven cost-benefit analysis to determine which sectors of ML-KEM's security must be guaranteed.

5.5 Example Scoring Systems

Although no single scoring system can be recommended, the following are two general scoring systems for organizations that favor confidentiality and availability or performance, respectively.

Variables in Example Scoring Systems:

ΔL: Leakage reduction (0 = none, 1 = fully reduced)

O_P: Performance overhead (fractional increase in cycles/memory/power consumption)

R_B: Break resistance (0 = fully breakable, 3 = unbreakable by first-order, higher-order, and deep-learning enhanced CPA and DPA attacks)

ΔT_L: Timing leakage mitigation (0 = fully mitigated, 1 = not mitigated)

A_S: Airtight Score (higher is better)

For meaningful assessment, each instance must be compared to a baseline defined by the organization, commonly an implementation of the scheme on a deliberately leaky board.

5.5.1 Example scoring system for high-confidentiality prioritization

Organizations that prioritize confidentiality will weigh break resistance and leakage mitigation higher than performance overhead to ensure security. Because of this, performance overhead will be weighted comparatively much lower. The following is a scoring system that takes these constraints into account:

$$A_S = (3 \times \Delta L) - (0.1 \times O_P) + (2 \times R_B^2) - (10 \times \Delta T_L)$$

This scoring system reflects a high-confidentiality prioritization by emphasizing leakage reduction and break resistance while minimizing the impact of performance overhead. The positive weight on ΔL rewards implementations that effectively reduce side-channel leakage, while the quadratic weight for R_B^2 further favors very strong break resistance. The small negative weight on O_P ensures performance penalties are considered but do not dominate the score. Finally, the negative weight on ΔT_L penalizes any residual timing leakage, aligning the score with security-critical requirements.

5.5.2 Example scoring system for high-availability prioritization

When Airtight is applied to implementations that focus on availability, performance overhead will be weighed heavily. Especially in lightweight applications, some countermeasures may be too computationally expensive to be feasible to implement. The following is a scoring system that prioritizes performance overhead as a metric without forfeiting security:

$$A_S = (3 \times \Delta L) - (2 \times O_P) + (2 \times R_B) - (7 \times \Delta T_L)$$

This equation reflects a high-availability prioritization by placing a stronger penalty on performance overhead (O_P), ensuring that computationally expensive countermeasures reduce the overall score. Leakage reduction (ΔL) remains positively weighted to maintain baseline security, while timing leakage mitigation (ΔT_L) is still penalized to discourage implementations with exploitable timing behavior. This balance allows lightweight or performance-sensitive applications to remain secure without sacrificing availability. The next section will discuss a strengths, weaknesses, opportunities, and threats (SWOT) analysis of Airtight.

5.5.3 Case study for an application of Airtight

To illustrate the applicability of Airtight, both scoring systems will be applied to a theoretical organizational improvement on the benchmark set in Chhetri (2026). Given Airtight's use case of evaluating organizational improvements in implementation, a baseline insecure implementation of ML-KEM is used as the baseline against which to compare a more secure implementation. Three methods will be used to increase the security of the implementation: decapsulation failure checks, constant-time algorithms, and changing to low-leakage hardware. Jati et al. (2024) state that decapsulation failure checking introduces approximately 5% overhead and Kolosick et al. (2025) state that constant-time execution adds another 5% overhead, on average. Combined, these security additions add 10.25% overhead by multiplying the two overhead costs together. Additionally, Ravi et al. (2024a) state that decapsulation failure checking can be defeated with higher-order CPA attacks, so implementing this protocol will increase R_B from 0 to 1. The implementation of constant-time algorithms will increase ΔT_L from 1 to 0. Finally, using low-leakage hardware will increase leakage detection to 0.9. These changes yield values of: $\Delta L = 0.9$, $O_P = 0.1025$, $R_B = 1$, and $\Delta T_L = 0$.

With these changes from the baseline to the more secure implementation, an organization may use Airtight to score the new implementation of ML-KEM. If the organization favors confidentiality, the new implementation's Airtight score will be 4.68975. If the organization prioritizes availability, the Airtight score

will be 4.495. The organizational leaders must then weigh the cost of changing to the new implementation against the Airtight score and determine whether to change or not. This scoring process enables reproducible and consistent scoring for ML-KEM implementations across varied organizational priorities.

5.6 SWOT Analysis

A systematic SWOT analysis enables organizations to assess how the Airtight framework aligns with their operational requirements and risk tolerance, particularly when deploying ML-KEM in environments vulnerable to side-channel attacks. By weighing strengths and opportunities against potential weaknesses and external threats, security professionals can better evaluate whether Airtight provides sufficient leakage resilience, resource feasibility, and long-term adaptability. The following SWOT analysis illustrates key internal and external factors that influence Airtight's effectiveness and may be tailored to reflect the mission constraints and security priorities of individual organizations.

5.6.1 Strengths

Airtight provides standardized and repeatable leakage detection methodologies that can be applied consistently across different hardware and implementation environments (Unger et al., 2022). Its modular phase structure allows evaluators to isolate leakage sources, integrate countermeasures incrementally, and compare mitigation effectiveness using quantifiable metrics. Furthermore, the framework's algorithm-agnostic design enables compatibility with multiple PQC schemes beyond ML-KEM, benefiting from worst-case to average-case security reductions established in lattice-based cryptography (Regev, 2009; Langlois and Stehlé, 2014). This supports long-term cryptographic agility and reuse within evolving DoD systems (U.S. Department of Defense, 2023).

5.6.2 Weaknesses

Airtight introduces overhead in the form of increased testing time, and specialized instrumentation, particularly for CPA and TVLA measurement environments (Eldib and Wang, 2014; Unger et al., 2022). Analyzing deep-learning–enhanced SCA traces may exceed the capabilities of organizations with limited technical resources, especially as recent attacks target ML-KEM's decapsulation and NTT operations (Primas, Pessl and Mangard, 2017; Kennaway et al., 2025). Additionally, the framework does not inherently eliminate leakage; instead, it depends on successful countermeasure integration, which may vary in effectiveness based on hardware and compiler constraints. Lastly, with an increased reliance on cloud computing, remote CPA attacks may be facilitated when side-channel data can be collected or inferred from shared hardware resources (Ravi et al., 2024b).

5.6.3 Opportunities

Airtight aligns with ongoing DoD efforts to transition to PQC and supports agency requirements for cryptographic agility and secure acquisition pathways (U.S. Department of Defense, 2023; Marmebro and Stenbom, 2024; NIST, 2024). By establishing consistent scoring criteria and certification levels, the framework could streamline procurement, reduce redundant testing, and enable shared evaluation baselines across services and contractors. Further opportunities arise in academia–DoD collaboration, where the framework could serve as a testbed for evaluating new SCA countermeasures or integrating secure hardware accelerators (Eldib and Wang, 2014; Ravi et al., 2024b).

5.6.4 Threats

Threats originate primarily from the rapid evolution of side-channel attack methodologies, particularly profiling attacks and deep-learning classifiers capable of bypassing masking or noise injection (Ravi et al., 2020; Kennaway et al., 2025). Attackers with access to advanced hardware or pre-characterized device models may reduce the effectiveness of standardized testing assumptions (Primas, Pessl and Mangard, 2017). Additionally, the pace of PQC research and potential revisions to ML-KEM parameters may outdate existing metrics or require framework adaptation, posing risks to long-term sustainability (NIST, 2024).

6. Conclusion

Organizations must prevent store-now, break-later attacks by transitioning to post-quantum cryptography early. The sooner a PQC scheme is standardized, the more secure communications will be. Ensuring cryptographic agility, resilience, and readiness is of paramount importance, and implementing ML-KEM represents a critical step toward this goal.

Before quantum computers achieve a high enough number of qubits to decrypt our current common encryption algorithms, PQC standards must be implemented. By implementing lattice-based cryptography, organizations can ensure their systems are more secure and resilient against quantum attacks. In tandem with PQC cryptography, comprehensive testing must be created and employed to strengthen security against side-channel attacks. By combining the use of secure cryptography with thorough testing, organizations can significantly enhance their security posture in an increasingly insecure environment. The Airtight framework represents a standardized, repeatable, and tailorable method for organizations to evaluate ML-KEM implementations, enabling confident, validated deployment in future secure ML-KEM communications.

Ethics declaration: Ethical clearance was not required for this research.

AI declaration: AI was used for this paper to convert between formatting styles and for spelling, punctuation, and grammar checking.

Disclaimer: The views expressed are those of the author and do not reflect the official policy or position of the US Air Force, Department of Defense or the US Government.

References

Arriaga, A., M. Barbosa, and S. Jarecki (2025). Tempo: ML-KEM to PAKE Compiler Resilient to Timing Attacks. Cryptology ePrint Archive.

Chhetri, R. (2026). “Benchmarking NIST-Standardised ML-KEM and ML-DSA on ARM Cortex-M0+: Performance, Memory, and Energy on the RP2040”. arXiv preprint arXiv:2603.19340.

Eldib, H. and C. Wang (July 2014). “Synthesis of Masking Countermeasures Against Side Channel Attacks”. In: International Conference on Computer Aided Verification. Cham: Springer International Publishing, pp. 114–130.

Jati, A. et al. (2024). “A configurable crystals-kyber hardware implementation with side-channel protection”. In: ACM transactions on embedded computing systems, 23(2), pp.1-25.

Kennaway, M. et al. (2025). “An Enhanced Two-Step CPA Side-Channel Analysis Attack on ML-KEM”. In: Proceedings of the 22nd International Conference on Security and Cryptography, pp. 263–274. DOI: 10.5220/0013638600003979.

Kolosick, M. et al. (2025). “Robust constant-time cryptography”. In: Proceedings of the ACM on Programming Languages, 9(PLDI), pp.1491-1515.

Langlois, A. and D. Stehlé (2014). “Worst-Case to Average-Case Reductions for Module Lattices”. In: Designs, Codes and Cryptography 75.3, pp. 565–599.

Marmebro, A. and K. Stenbom (2024). Investigation of Post-Quantum Cryptography (FIPS 203 & 204) Compared to Legacy Cryptosystems, and Implementation in Large Corporations.

National Institute of Standards and Technology (2024). Module-Lattice-Based Key-Encapsulation Mechanism Standard. Federal Information Processing Standard 203. Washington, DC: U.S. Department of Commerce. DOI: 10.6028/NIST.FIPS.203.

Nehal, S., M. Farhan, and Y. Sunad (2020). “Quantum Cryptography—Breaking RSA Encryption Using Quantum Computing with Shor’s Algorithm”. In: International Journal For Technological Research In Engineering 8.1, pp. 22–27.

Primas, R., P. Pessl, and S. Mangard (Aug. 2017). “Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption”. In: International Conference on Cryptographic Hardware and Embedded Systems. Cham: Springer International Publishing, pp. 513–533.

Ravi, P. et al. (2020). “Generic Side-Channel Attacks on CCA-Secure Lattice-Based PKE and KEMs”. In: IACR Transactions on Cryptographic Hardware and Embedded Systems, pp. 307–335.

Ravi, P. et al. (2024a). “Defeating Low-Cost Countermeasures Against Side-Channel Attacks in Lattice-Based Encryption”. In: IACR Transactions on Cryptographic Hardware and Embedded Systems.

Ravi, P. et al. (2024b). “Side-channel and Fault-injection attacks over Lattice-based Post-quantum Schemes (Kyber, Dilithium): Survey and New Results”. In: ACM Trans. Embed. Comput. Syst. 23.2. ISSN: 1539-9087. URL:https://doi.org/10.1145/3603170.

Regev, Oded (2009). “On Lattices, Learning with Errors, Random Linear Codes, and Cryptography”. In: Journal of the ACM 56.6, pp. 1–40.

U.S. Department of Defense, Chief Information Officer (Sept. 14, 2023). DoD CIO Memo on Migration to Stronger Cryptographic Algorithms. DoD Cyber Exchange. URL:https://www.cyber.mil/pki-pke/ cryptographic-modernization/.

Unger, W. et al. (May 10, 2022). TVLA, Correlation Power Analysis and Side-Channel Leakage Assessment Metrics. URL:https://csrc.nist.gov/csrc/media/Events/2022/lightweight-cryptography-workshop-2022/documents/papers/tvla- correlation-power-analysis-and-side-channel-leakage-assessment-metrics.pdf.

Zhou, YongBin and DengGuo Feng (2005). Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing. Cryptology ePrint Archive, Paper 2005/388. URL: https://eprint.iacr.org/2005/388.