Design and Performance Evaluation of Post-Quantum Authentication for Embedded Systems: A Case Study on PIV*

Emmanuelle Dottax and Rina Zeitoun

IDEMIA Secure Transactions, France
firstname.lastname@idemia.com

Abstract. As the transition to post-quantum cryptography accelerates, security protocols must evolve to resist quantum threats while remaining practical, particularly on constrained devices where memory, bandwidth, and performance are limited. We consider the NIST Personal Identity Verification (PIV) system, where smart cards rely on digital signatures for authentication. Since post-quantum signatures introduce substantial computational and memory overhead, whereas post-quantum Key Encapsulation Mechanisms (KEMs) are generally lighter, we investigate KEM-based alternatives for authentication and assess the migration of secure messaging to post-quantum primitives. We propose post-quantum variants of the PIV authentication and secure messaging protocols and implement both signature-based and KEM-based approaches on a real smart card platform. We evaluate their computational and communication costs in a realistic embedded setting and present detailed performance metrics that enable assessing the impact of post-quantum migration across different hardware and communication configurations. Our results show that KEM-based authentication significantly reduces execution time and transmitted data compared to post-quantum signature-based designs, while KEM-based post-quantum secure messaging incurs moderate overhead compared to its classical counterpart. These findings highlight KEM-based authentication as a practical migration strategy for post-quantum secure embedded systems.

Keywords: Post-Quantum Cryptography · Authentication · Secure Channel · Protocol Design · Embedded Device · PIV

1 Introduction

The transition to post-quantum (PQ) cryptography is now underway, and existing protocols must be adapted accordingly. Several PQ standards are already emerging, but their integration into embedded applications, such as those relying on secure elements, poses specific challenges. Secure elements operate under tight computational, memory, and bandwidth constraints, while PQ algorithms, and especially PQ signatures, tend to be significantly more demanding than their classical counterparts. Beyond performance constraints, embedded devices such as smart cards operate in hostile environments: they

* This is an extended version of [DZ26]

are physically accessible to adversaries and must therefore withstand implementation-level attacks, including side-channel and fault injection attacks. As a consequence, cryptographic implementations in secure elements must incorporate dedicated countermeasures, which may further increase code size, memory usage, and execution time. Taken together, these constraints make the migration of existing mechanisms in a quantum-resistant setting a non-trivial task. It often requires dedicated design choices rather than a straightforward drop-in replacement.

In this work, we focus on the NIST Personal Identity Verification (PIV) standard, which specifies secure and reliable identification mechanisms for U.S. federal employees and contractors. It is implemented to control physical access to facilities as well as logical access to information systems and applications. At the core of this system are PIV cards, integrated circuits used for binding identities to credentials. They are involved in cardholder authentication mechanisms with varying characteristics, some providing authentication-only, while more advanced ones establish a secure channel between the PIV card and the system. Both use-cases need attention to reach quantum-resistance while maintaining existing security features and usability.

Authentication-only. Authentication is the cornerstone of the overall security architecture, as it establishes trust between the embedded device and the infrastructure. PIV defines two protocols that play a central role in this context, PKI-AUTH and PKI-CAK. Both are based on asymmetric signatures and certificates, and a "challenge-response" type of protocol. PKI-AUTH extends PKI-CAK by requiring, prior to the card signature, the verification of a user-entered PIN, thereby ensuring cardholder authentication. Both are foundational building blocks used across a wide range of PIV deployments.

In the current PIV specifications, primarily FIPS 201-3 [NIS22] and SP 800-73 [NIS24b,NIS24c,NIS24d], card authentication makes use of RSA or ECDSA to sign a challenge sent by the relying system. These algorithms offer relatively fast execution and compact key and signature sizes, making them well suited for embedded authentication in the classic setting. In the post-quantum setting, however, signature schemes incur significantly higher computational cost, larger RAM requirements, and substantially increased key and signature sizes. Moreover, among the currently standardized PQ signature schemes, only one appears realistically deployable on smart cards, and even then under specific constraints that we discuss below. Such limitations can push embedded devices to their limit boundaries and negatively impact the user experience.

These limitations motivate a re-evaluation of PKI-AUTH and PKI-CAK protocols in the post-quantum setting. Generally, to achieve authentication, two directions can be considered: using digital signatures, like current PIV implementations, or relying on a key establishment mechanism. With PQ algorithms, the second path becomes especially appealing for constrained devices. Indeed, PQ KEMs are generally lighter than PQ signature schemes in several aspects: execution time, RAM consumption and data size. In this work we explore both options and evaluate the impact on security and performances.

Secure Messaging. Another essential building block of the PIV ecosystem is the secure messaging mechanism, referred to as SM-AUTH. This protocol is notably used within

the OCC-AUTH protocol [NIS24b] to support biometric-based authentication, where it secures the exchange involved in the biometric verification process.

Unlike PKI-AUTH and PKI-CAK which rely on signatures, SM-AUTH relies on a key-agreement protocol. In current specifications [NIS24c], secure messaging is established through an Elliptic Curve Diffie-Hellman (ECDH) key exchange between the card and the relying system. The resulting shared secret is then used to derive session keys that protect subsequent communications in confidentiality and integrity. While efficient, ECDH does not provide resistance against quantum adversaries. Migrating SM-AUTH to the post-quantum setting naturally suggests replacing classical ECDH with a KEM-based construction, which preserves the overall key-establishment structure while achieving quantum resistance. We analyze the impact of this modification on the protocol's computational and communication performance.

Related work. Several works have investigated the use of KEMs as an alternative to digital signatures when adapting existing protocols to the post-quantum setting. For instance, KEMTLS [SSW20,SSW21,CFHS+21] proposes a variant of the TLS handshake where signatures are replaced with KEMs to reduce the required bandwidth (KEM ciphertexts being shorter than signatures). A generic approach of replacing signatures by KEMs is presented in [BdKM23], which provides a flexible framework. While its design is intentionally generic rather than optimized for a particular protocol, it highlights important considerations when making this transitioning. Other protocols have undergone PQ adaptations using KEMs. PQNoise [ADH+22], PQ-WireGuard [HNS+21], the Muckle# variant of Muckle+ [BSP+25], and the KEM-based X3DH proposal [BFG+21] all follow this general direction. These papers provide insights into the use of KEMs in modern protocol design, although they do not include practical comparisons between signature-based and KEM-based post-quantum variants, which is an aspect of particular interest for our study. Moreover, all the above implementations target desktop-class environments rather than embedded platforms, whose stringent resource constraints can significantly impact design and performance considerations.

Further closely related contributions are the studies in [FvdHM+23], [BDD22] and [BDG25], as they likewise target an embedded setting. The scope of [FvdHM+23] is specific to the Extended Access Control protocol used in ICAO passports. Though some results could be relevant to understand the advantages of using a KEM in replacement of a signature, its Kyber implementation is significantly slower than expected, which limits the reliability of their performance results. The study in [BDG25] investigates adaptations of protocols used in the context of eSIMs, while [BDD22] focuses on EMV protocols for banking applications. However, neither work provides a practical comparison between signature-based and KEM-based variants in the post-quantum setting, leaving open the question of their relative efficiency in constrained environments and offering little guidance as to whether modifying credential formats to move from signature-based to KEM-based constructions is worthwhile.

Our Contributions. In this work, we explore post-quantum adaptations of the PIV authentication protocols PKI-AUTH and PKI-CAK, as well as of the secure messaging mechanism SM-AUTH. In particular, we investigate KEM-based constructions as alternatives to signature-based designs.

We make explicit the implications of moving from a signature-based setting to a KEM-based one, in terms of credential formats, protocol flows, and, in some cases, security properties. We also analyze potential advantages in terms of implementation simplification and code mutualization.

To obtain a clear assessment of performance, we implement the proposed constructions on a real smart card platform, using implementations hardened against side-channel and fault attacks. We provide concrete measurements of computational and communication costs in a realistic embedded setting. In particular, we report separate evaluation of on-card computation and data transmission, enabling fine grained performance estimation under different communication conditions (e.g., contact vs. contactless).

Our results show that, for PKI-AUTH and PKI-CAK in a post-quantum setting, KEM-based authentication clearly outperforms the signature-based variant in terms of card-side execution time, while also reducing the amount of exchanged data and, consequently, communication latency. For SM-AUTH, replacing ECDH with a post-quantum KEM provides quantum-resistance with moderate computational and communication overhead, making it a practical solution for constrained environments.

Although our study focuses on the PIV ecosystem, the design principles developed in this work are not PIV-specific. The proposed KEM-based approach can be transported to other authentication and key-establishment protocols facing similar post-quantum migration challenges, especially in constrained embedded environments.

Paper Organization. The remainder of the paper is organized as follows. Section 2 recalls the original PKI-AUTH, PKI-CAK and SM-AUTH protocols. Section 3 presents our proposed quantum-resistant variants. Section 4 reports detailed implementation and performance results for the different approaches. Finally, Sect. 5 concludes the paper.

2 Original PIV Protocols

We consider the different card-based protocols specified by PIV. Relevant information is found in several documents by NIST: the main standard FIPS 201-3 [NIS22] provides high-level information on protocols, and is completed by the SP 800-73 serie [NIS24b,NIS24c,NIS24d] for information on card implementation and SP 800-78 for cryptographic details [NIS24a].

On one side are protocols providing authentication only, namely PKI-AUTH and PKI-CAK. Both protocols allow authentication of the card, using asymmetric cryptography and relying on certificates. Their main distinction lies in the additional security property offered by PKI-AUTH: in this case, the card must be *activated* through cardholder verification (using PIN validation) before authenticating, thereby enforcing two-factor authentication. Together, these two protocols provide graduated levels of authentication assurance for physical and logical access within the PIV ecosystem: while PKI-CAK provides "Some confidence in the asserted identity", PKI-AUTH offers "High confidence in the asserted identity". Organizations can therefore select the authentication mechanism that matches the security requirements of the resource being accessed.

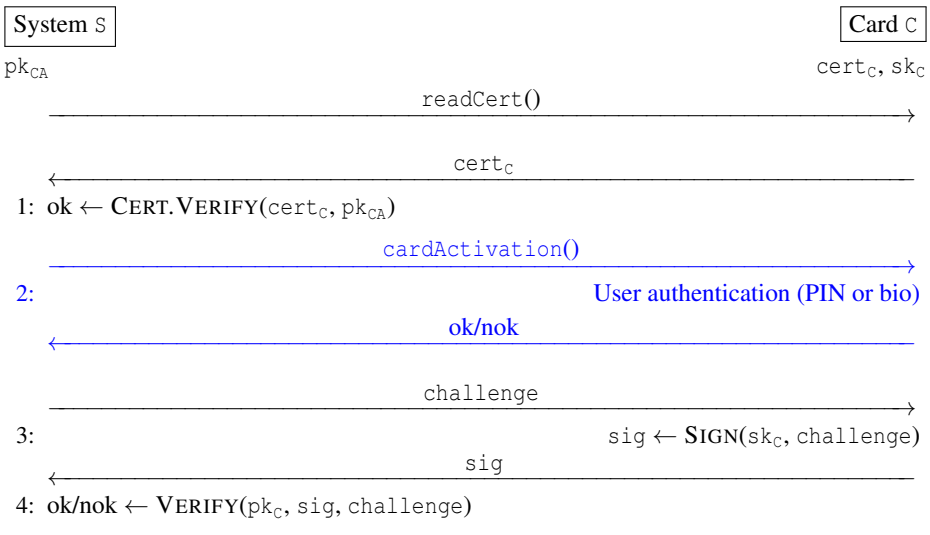
On the other side, an optional protocol SM-AUTH supports the establishment of a secure messaging in addition to authentication of the card. It enables secure exchange of data and commands, for instance when using a contactless interface.

In the following, we present these protocols as specified today, authentication ones in Sect. 2.1 and the secure messaging in Sect. 2.2. The off-card entity, denoted System, can be a terminal, a reader, or a distant party like a server, depending on the use-case.

2.1 Authentication Protocols

The PKI-AUTH and PKI-CAK mechanisms, whose message flows are shown in Protocol 1 and follow the diagrams in [NIS24c], provide cryptographic authentication of the PIV card. In both protocols, the system begins by verifying the card's certificate $\texttt{cert}_C$. In PKI-AUTH, the cardholder is additionally required to enter a PIN, providing user authentication on top of card authentication. After these preliminary checks, the system challenges the card to sign a nonce `challenge`, typically using RSA or ECDSA [NIS23]. The system can then verify the resulting signature `sig` using the public key from the card's certificate $\texttt{pk}_C$. Since only the card possesses the corresponding private key $\texttt{sk}_C$, a valid signature proves the card's authenticity.

Protocol 1 PIV authentication protocols PKI-CAK and PKI-AUTH. Commands in blue are mandatory for PKI-AUTH.



2.2 Secure Messaging Protocol

We now consider the PIV secure messaging protocol SM-AUTH whose simplified message flow is shown in Protocol 2, following [NIS24b] and [NIS24c]. SM-AUTH is an

optional but recommended mechanism that enables protected communication with a PIV card. For readability, some details of the original specification have been omitted, but these simplifications do not affect the aspects relevant to our post-quantum adaptation.

In this protocol, the system and the card first establish a shared secret using an Elliptic-Curve Diffie-Hellman (ECDH) exchange: the sytem generates an ephemeral key pair ($\mathtt{esk_S}$, $\mathtt{epk_S}$), while the card relies on a static key pair $\mathtt{sk_C}$, and both derive the same shared value $\mathtt{Z}$ through the cofactor variant of ECDH, denoted as ECC-CDH following NIST notation. This shared secret, together with additional inputs such as a freshly generated nonce $\mathtt{N_C}$ and the first 16 bytes of the system's ephemeral public key $\mathtt{epk_S}$, is fed into a key-derivation function KDF to derive the session keys $\mathtt{K_{CFRM}}$, $\mathtt{K_{MAC}}$, $\mathtt{K_{ENC}}$, and $\mathtt{K_{RMAC}}$.

These session keys are then used to protect subsequent exchanges. In particular, the session key $\mathtt{K_{CFRM}}$ serves as a MAC key: the card computes a CMAC [NIS05] over well-defined data, including the full system's ephemeral public key $\mathtt{epk_S}$. The resulting value $\mathtt{AuthC}$ is sent to the system, which recomputes it and verifies equality. Successful verification authenticates the card to the system.

3 Quantum-Resistant Versions

We explore quantum-safe variants of the PIV protocols. While major European agencies, such as the French ANSSI [ANS23] and the German BSI [BSI26], recommend a hybrid approach during the migration phase, NIST does not actively promote this strategy (although it remains acceptable [MPR+24]). Consequently, we focus exclusively on purely post-quantum constructions.

Hybrid versions based on our proposals are nevertheless straightforward to derive. Moreover, the performance results reported in Sect. 4 remain largely applicable in this context, as the dominant overhead—both in computation and communication size—is driven by the post-quantum components.

3.1 Post-Quantum Authentication Protocols

In their original form, the PKI-AUTH and PKI-CAK protocols rely on classical RSA or ECDSA for the signature operation. As both cryptographic algorithms are vulnerable to quantum adversaries, PKI-AUTH and PKI-CAK, in their current form, do not offer post-quantum security. In the following, we propose post-quantum variants of these protocols using two approaches.

Post-Quantum Authentication Using Signature. The first approach consists in directly replacing the classical signature algorithm with a post-quantum scheme in Protocol 1. In practice, the signature primitive SIGN is instantiated with a PQ signature algorithm, thereby preserving the overall structure and flow of the original protocol while making it resistant to quantum adversaries. Of course, the card certificate $\mathtt{cert_C}$ must also be issued using a post-quantum signature scheme.

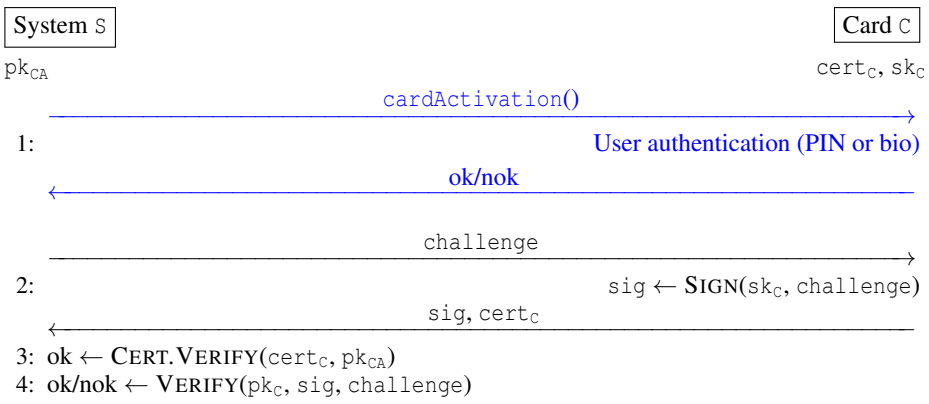
Protocol 2 PIV secure messaging protocol SM-AUTH (simplified).

System S		Card C
$\mathtt{pk}_{\mathtt{CA}}$		$\mathtt{cert}_{\mathtt{C}}$, $\mathtt{sk}_{\mathtt{C}}$
1: $\mathtt{CB}_{\mathtt{S}} \leftarrow$ 0x00		
2: $(\mathtt{esk}_{\mathtt{S}}, \mathtt{epk}_{\mathtt{S}}) \leftarrow$ ECKG()		
	$\mathtt{CB}_{\mathtt{S}}, \mathtt{ID}_{\mathtt{S}}, \mathtt{epk}_{\mathtt{S}} \longrightarrow$	
3:		$\mathtt{ID}_{\mathtt{C}} \leftarrow T_8(\text{SHA-256}(\mathtt{cert}_{\mathtt{C}}))$
4:		$\mathtt{CB}_{\mathtt{C}} \leftarrow \mathtt{CB}_{\mathtt{S}}$ & 0xF0
5:		Check $\mathtt{CB}_{\mathtt{C}} =$ 0x00
6:		Check $\mathtt{epk}_{\mathtt{S}}$ is a valid public key
7:		$\mathtt{Z} \leftarrow$ ECC-CDH($\mathtt{sk}_{\mathtt{C}}$, $\mathtt{epk}_{\mathtt{S}}$)
8:		Generate nonce $\mathtt{N}_{\mathtt{C}}$
9:		$\mathtt{OtherInfo} \leftarrow$ F($\mathtt{ID}_{\mathtt{S}}$, $\mathtt{ID}_{\mathtt{C}}$, $\mathtt{CB}_{\mathtt{S}}$, $\mathtt{CB}_{\mathtt{C}}$, $\mathtt{N}_{\mathtt{C}}$, $T_{16}(\mathtt{epk}_{\mathtt{S}})$)
10:		$\mathtt{K}_{\mathbf{CFRM}} \| \mathtt{K}_{\mathbf{MAC}} \| \mathtt{K}_{\mathbf{ENC}} \| \mathtt{K}_{\mathbf{RMAC}} \leftarrow$ KDF(Z, len, OtherInfo)
11:		Zeroize Z
12:		$\mathtt{AuthC} \leftarrow$ CMAC($\mathtt{K}_{\mathbf{CFRM}}$, label $\| \mathtt{ID}_{\mathtt{C}} \| \mathtt{ID}_{\mathtt{S}} \| \mathtt{epk}_{\mathtt{S}}$)
13:		Zeroize $\mathtt{K}_{\mathbf{CFRM}}$
	$\longleftarrow \mathtt{CB}_{\mathtt{C}}, \mathtt{N}_{\mathtt{C}}, \mathtt{AuthC}, \mathtt{cert}_{\mathtt{C}}$	
14: Check $\mathtt{CB}_{\mathtt{C}} =$ 0x00		
15: ok $\leftarrow$ CERT.VERIFY($\mathtt{cert}_{\mathtt{C}}$, $\mathtt{pk}_{\mathtt{CA}}$)		
16: $\mathtt{ID}_{\mathtt{C}} \leftarrow T_8(\text{SHA-256}(\mathtt{cert}_{\mathtt{C}}))$		
17: Extract $\mathtt{pk}_{\mathtt{C}}$ from $\mathtt{cert}_{\mathtt{C}}$		
18: $\mathtt{Z} \leftarrow$ ECC-CDH($\mathtt{esk}_{\mathtt{S}}$, $\mathtt{pk}_{\mathtt{C}}$)		
19: Zeroize $\mathtt{esk}_{\mathtt{S}}$		
20: $\mathtt{OtherInfo} \leftarrow$ F($\mathtt{ID}_{\mathtt{S}}$, $\mathtt{ID}_{\mathtt{C}}$, $\mathtt{CB}_{\mathtt{S}}$, $\mathtt{CB}_{\mathtt{C}}$, $\mathtt{N}_{\mathtt{C}}$, $T_{16}(\mathtt{epk}_{\mathtt{S}})$)		
21: $\mathtt{K}_{\mathbf{CFRM}} \| \mathtt{K}_{\mathbf{MAC}} \| \mathtt{K}_{\mathbf{ENC}} \| \mathtt{K}_{\mathbf{RMAC}} \leftarrow$ KDF(Z, len, OtherInfo)		
22: Zeroize Z		
23: Check that $\mathtt{AuthC} =$ CMAC($\mathtt{K}_{\mathbf{CFRM}}$, label $\| \mathtt{ID}_{\mathtt{C}} \| \mathtt{ID}_{\mathtt{S}} \| \mathtt{epk}_{\mathtt{S}}$)		
24: Zeroize $\mathtt{K}_{\mathbf{CFRM}}$		

This approach minimizes changes to the protocol logic: the command structure and message flow remain essentially unchanged, apart from the increased size of cryptographic data. The type of credentials is also preserved, as authentication still relies on digital signatures. Assuming the PQ signature scheme provides security guarantees comparable to those of the replaced classical primitive, the overall security properties of the protocol remain unchanged. As a result, this migration path requires limited development effort and facilitates integration into existing infrastructures.

If reducing the number of exchanges is desirable, the card certificate $\mathtt{cert}_{\mathtt{C}}$ may be transmitted together with the authentication signature rather than being retrieved in advance. The system can then verify both the certificate and the signature upon reception, reducing the protocol to a single round trip, as illustrated in Protocol 3. This optimization is applicable to both the classical and post-quantum variants, as it does not alter the underlying authentication mechanism or its security properties.

Protocol 3 Single-Round-Trip PIV Authentication Protocols PKI-CAK and PKI-AUTH. Commands in blue are mandatory for PKI-AUTH.



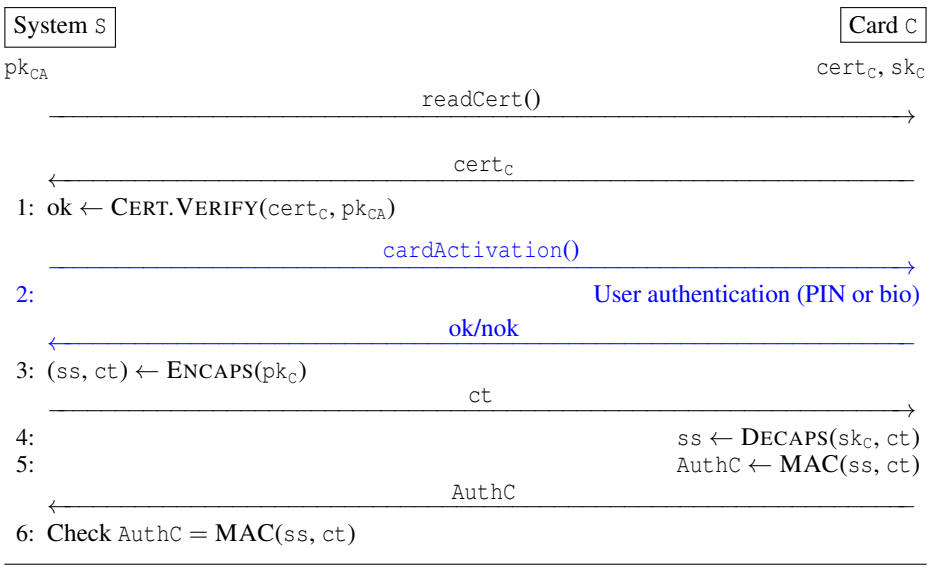
Post-Quantum Authentication Using KEM. The second approach replaces the signature-based authentication mechanism with a KEM-based construction. Instead of proving possession of a private signing key through a signature, the card demonstrates knowledge of a secret key associated with a key-encapsulation mechanism. Authentication is then derived from the successful establishment of a shared secret.

As already mentioned, such a transformation has been investigated in several works on post-quantum protocol design. Our proposal therefore builds upon established design principles and inherits the corresponding security analyses developed in prior literature. In particular, our design follows the one outlined in NIST document SP 800-227 [NIS25, Sect. 5.2.2, Fig. 8], which specifies a KEM-based authentication pattern relying on key confirmation.

The resulting design is depicted in Protocol 4. After verification of the card's certificate, the system encapsulates the card's public key pk_C, producing a public ciphertext ct and a shared secret ss. Upon receiving ct, the card decapsulates it using its private key sk_C to recover ss. A subsequent key-confirmation step ensures that both parties derived the same shared secret: each computes a MAC value AuthC over ct using ss as key. Successful verification assures the system that the card possesses the correct private key sk_C, thereby achieving authentication.

This modification impacts several aspects of the design. Credential formats must be adapted, as the card's key pair (sk_C, pk_C) is now a KEM key pair rather than a signature key pair. Nevertheless, there is no technical barrier to deploying such certificates, which can be generated, verified, and provisioned using standard processes. The protocol flow and command structure may require adjustments to accommodate encapsulation and decapsulation operations. In particular, whereas signature-based authentication can be performed in a single challenge-response exchange as depicted in Protocol 3, a KEM-based construction requires the system to obtain the card's public key before encapsulation, potentially introducing an additional communication step.

Protocol 4 Post-quantum KEM-based authentication protocols PKI-CAK and PKI-AUTH. Commands in blue are mandatory for PKI-AUTH.



Moreover, the security analysis must be revisited, as authentication no longer relies on the verification of a digital signature, but on key confirmation derived from a shared secret established during protocol execution. In the original signature-based design, authentication is tied to possession of the card's long-term signing key. A valid signature can only be produced by the legitimate card and therefore constitutes a cryptographic proof of participation. As a result, a protocol transcript containing the challenge and the corresponding signature may serve as transferable evidence that the card holder participated in the exchange, providing *non-repudiation* [NISnd]. In contrast, the KEM-based construction authenticates the card through knowledge of the shared secret established during the protocol. While this effectively achieves authentication during the interaction, the resulting transcript no longer constitutes a proof of participation. Indeed, encapsulation and shared-secret derivation can be performed by any party using the card's public key, allowing complete protocol transcripts to be simulated without access to the card's private key. Consequently, the protocol provides *deniability* rather than non-repudiation, a property that may be desirable in privacy-sensitive deployments.

This change in the authentication model has another consequence. Since authentication is now tied to possession of the established shared secret rather than to the card's long-term private key, an adversary capable of accessing the system's memory during protocol execution may impersonate the card. More precisely, an attacker able to read the RAM of the system can recover the shared secret ss computed at Step 3 of Protocol 4. Knowledge of ss is sufficient to derive the authentication value AuthC expected by the terminal, allowing the attacker to successfully authenticate as the card without compromising its private key. This is in contrast with the signature-based construc-

tion, where generating the authentication token requires access to the card's signing key. Therefore, the KEM-based approach shifts part of the security assumptions from the protection of a long-term secret to the protection of ephemeral protocol state. In practice, this does not significantly alter the threat model, as the system is already expected to protect its internal state and security-critical computations. Nevertheless, this distinction should be taken into account when evaluating the security properties of the protocol.

More generally, while the proposed approach entails greater design and development effort, it may offer significant advantages in constrained environments. In particular, KEM-based constructions can reduce computational costs and data sizes compared to post-quantum signature schemes, potentially improving performance and user experience on resource-limited embedded platforms (see Sect. 4).

However, this approach may be unsuitable when authentication must comply with externally standardized protocols, such as TLS [Res18], which mandates the use of signatures for authentication.

3.2 Post-Quantum Secure Messaging Protocol

At the heart of the secure messaging protocol lies the establishment of a shared secret from which session keys are derived. In the original design, this is achieved using ECDH. Since ECDH is vulnerable to quantum attacks, the resulting secure messaging session inherits this limitation. We therefore propose to replace the ECDH mechanism with a post-quantum alternative in which the shared secret is established through a KEM encapsulation/decapsulation procedure, while preserving the same MAC-based authentication structure. This substitution is natural: since ECDH is used solely to derive a shared secret, a KEM offers the direct post-quantum analogue for fulfilling this role. The resulting design is shown in Protocol 5.

Adopting a KEM-based construction slightly modifies the message flow. In particular, whereas the original ECDH-based protocol completes in a single round trip, the KEM-based variant requires an additional card-to-system exchange to provide the card's certificate $\texttt{cert}_\texttt{C}$ (and thus its public key $\texttt{pk}_\texttt{C}$) before encapsulation can be performed. This additional step is necessary because KEM encapsulation, unlike ECDH key agreement, requires prior knowledge of the recipient's public key. Once the public key is available, the system performs a KEM encapsulation operation to produce a ciphertext $\texttt{ct}$ and a shared secret $\texttt{ss}$. The ciphertext is then sent to the card, which uses its private key $\texttt{sk}_\texttt{C}$ to decapsulate and recover the same secret value $\texttt{ss}$, an operation that can only be performed by a legitimate card.

From this shared secret, both parties derive the session keys by adapting the KDF inputs: the first 16 bytes of the ECDH public key $\texttt{epk}_\texttt{S}$ are replaced by the first 16 bytes of the KEM ciphertext $\texttt{ct}$, which constitutes the public output of the encapsulation. The derived session key $\texttt{K}_{\texttt{CFRM}}$ is then used to compute a MAC $\texttt{AuthC}$ over appropriately updated inputs. In particular, the ephemeral ECDH public key $\texttt{epk}_\texttt{S}$ is replaced by the KEM ciphertext $\texttt{ct}$, and the card-generated nonce $\texttt{N}_\texttt{C}$ is explicitly included. This design follows the recommendations of SP 800-227 [NIS25, Sect. 4.4.1], which require the use of two sources of freshness in the key-confirmation process: one provided by the system (the ciphertext $\texttt{ct}$) and one provided by the card (the nonce $\texttt{N}_\texttt{C}$).

The validation of these MACs provides explicit key confirmation: successful verification demonstrates that the card recovered the correct shared secret, thereby ensuring both secure messaging and card authentication in a quantum-resistant manner.

Protocol 5 Post-quantum KEM-based PIV secure messaging protocol SM-AUTH (simplified).

System S		Card C
$\mathtt{pk}_{\mathtt{CA}}$		$\mathtt{cert}_{\mathtt{C}}$, $\mathtt{sk}_{\mathtt{C}}$
	readCert() $\longrightarrow$	
	$\longleftarrow$ $\mathtt{cert}_{\mathtt{C}}$	
ok $\leftarrow$ CERT.VERIFY($\mathtt{cert}_{\mathtt{C}}$, $\mathtt{pk}_{\mathtt{CA}}$)		
Extract $\mathtt{pk}_{\mathtt{C}}$ from $\mathtt{cert}_{\mathtt{C}}$		
($\mathtt{ss}$, $\mathtt{ct}$) $\leftarrow$ ENCAPS($\mathtt{pk}_{\mathtt{C}}$)		
$\mathtt{CB}_{\mathtt{S}} \leftarrow$ 0x00		
	$\mathtt{CB}_{\mathtt{S}}$, $\mathtt{ID}_{\mathtt{S}}$, $\mathtt{ct}$ $\longrightarrow$	
		$\mathtt{ID}_{\mathtt{C}} \leftarrow T_8$(SHA-256($\mathtt{cert}_{\mathtt{C}}$))
		$\mathtt{CB}_{\mathtt{C}} \leftarrow \mathtt{CB}_{\mathtt{S}}$ & 0xF0
		Check $\mathtt{CB}_{\mathtt{C}} =$ 0x00
		$\mathtt{ss} \leftarrow$ DECAPS($\mathtt{sk}_{\mathtt{C}}$, $\mathtt{ct}$)
		Generate nonce $\mathtt{N}_{\mathtt{C}}$
		$\mathtt{OtherInfo} \leftarrow$ F($\mathtt{ID}_{\mathtt{S}}$, $\mathtt{ID}_{\mathtt{C}}$, $\mathtt{CB}_{\mathtt{S}}$, $\mathtt{CB}_{\mathtt{C}}$, $\mathtt{N}_{\mathtt{C}}$, T_{16}($\mathtt{ct}$))
		$\mathtt{K}_{\mathbf{CFRM}} \| \mathtt{K}_{\mathbf{MAC}} \| \mathtt{K}_{\mathbf{ENC}} \| \mathtt{K}_{\mathbf{RMAC}} \leftarrow$ KDF($\mathtt{ss}$, len, $\mathtt{OtherInfo}$)
		Zeroize $\mathtt{ss}$
		$\mathtt{AuthC} \leftarrow$ MAC($\mathtt{K}_{\mathbf{CFRM}}$, label $\| \mathtt{ID}_{\mathtt{C}} \| \mathtt{ID}_{\mathtt{S}} \| \mathtt{N}_{\mathtt{C}} \| \mathtt{ct}$)
		Zeroize $\mathtt{K}_{\mathbf{CFRM}}$
	$\mathtt{CB}_{\mathtt{C}}$, $\mathtt{N}_{\mathtt{C}}$, $\mathtt{AuthC}$ $\longleftarrow$	
Check $\mathtt{CB}_{\mathtt{C}} =$ 0x00		
$\mathtt{ID}_{\mathtt{C}} \leftarrow T_8$(SHA-256($\mathtt{cert}_{\mathtt{C}}$))		
$\mathtt{OtherInfo} \leftarrow$ F($\mathtt{ID}_{\mathtt{S}}$, $\mathtt{ID}_{\mathtt{C}}$, $\mathtt{CB}_{\mathtt{S}}$, $\mathtt{CB}_{\mathtt{C}}$, $\mathtt{N}_{\mathtt{C}}$, T_{16}($\mathtt{ct}$))		
$\mathtt{K}_{\mathbf{CFRM}} \| \mathtt{K}_{\mathbf{MAC}} \| \mathtt{K}_{\mathbf{ENC}} \| \mathtt{K}_{\mathbf{RMAC}} \leftarrow$ KDF($\mathtt{ss}$, len, $\mathtt{OtherInfo}$)		
Zeroize $\mathtt{ss}$		
Check that $\mathtt{AuthC} =$ MAC($\mathtt{K}_{\mathbf{CFRM}}$, label $\| \mathtt{ID}_{\mathtt{C}} \| \mathtt{ID}_{\mathtt{S}} \| \mathtt{N}_{\mathtt{C}} \| \mathtt{ct}$)		
Zeroize $\mathtt{K}_{\mathbf{CFRM}}$		

Both the classical and the PQ secure messaging protocols are based on a key-establishment mechanism. Consequently, while authentication is achieved during the execution of the protocol, the resulting transcript does not constitute transferable proof of participation since a complete protocol transcript can be generated without access to the card's private key. The transition to a KEM-based design therefore preserves the deniability property already present in the classical secure messaging protocol.

We stress that in the post-quantum setting, the use of CMAC is not mandatory: any MAC algorithm providing equivalent security guarantees may be used. Likewise, the key-derivation function (KDF), typically instantiated using a hash function such as

SHA-2, must be selected and parameterized to match the targeted PQ security level, ensuring that the derived keys provide a level of security consistent with that of the underlying KEM.

4 Performance Analysis

This section evaluates the performance of the protocols in both classical and post-quantum settings. In the constrained environment of a secure element, execution time and communication overhead are critical, as they directly affect user experience. The evaluation relies on implementations deployed on a production-grade smart card platform embedding a secure element designed for banking and identity applications, and running at 50 MHz. Cryptographic primitives incorporate countermeasures against side-channel and fault attacks [CGMZ22,CGMZ23,CGTZ23,CGL+24], providing a realistic assessment under practical deployment constraints.

To improve clarity and portability of the results, execution times and communication costs are reported separately in distinct tables. This separation makes it possible to exploit the measured computation times independently of the communication assumptions, and to adapt the communication latency to different transmission data rates or deployment scenarios. It thereby supports a finer-grained evaluation in alternative contexts, where transmission speeds, channel characteristics, or system constraints may differ.

Furthermore, for card-side computations, we systematically report normalized values relative to the fastest configuration. Absolute timings may vary significantly across platforms, due for instance to differences in CPU performance, memory constraints, or the level of protection against physical attacks. Relative ratios therefore provide a more transferable measure of the practical overhead induced by the post-quantum transition, while improving clarity and comparability.

Section 4.1 is dedicated to authentication protocols. For the classical setting, we consider Protocol 1 instantiated with either RSA or ECDSA, as specified by NIST. In the post-quantum setting, we evaluate both alternatives introduced in Sect. 3.1: the signature-based variant Protocol 1 instantiated with a PQ signature scheme and the KEM-based variant Protocol 4 instantiated with a PQ KEM. For the signature-based variant, several schemes are included in the NIST (soon-to-be finalized) standards. Stateful signatures schemes [NIS20] are not suitable for our use-case, as the number of signatures produced by the card cannot be strictly bounded. SLH-DSA [NIS24g] and Falcon [FHK+20] are impractical for smart card implementations: SLH-DSA entails prohibitively high signature-generation costs [DN24], while Falcon is challenging to implement securely without excessive RAM consumption [EFG+22]. Consequently, ML-DSA [NIS24e] is the only practical candidate in our setting. For the KEM-based variant, the NIST selection includes ML-KEM [NIS24f] and HQC [HQC25]. As ML-KEM offers better performance in terms of both execution time and data size, we restrict our evaluation to this scheme.

Section 4.2 tackles the secure messaging use-case. For the classical setting we consider ECDH, as specified by NIST. For the post-quantum setting, here again we consider ML-KEM.

In all post-quantum configurations, the signature algorithm used to issue the card certificate remains flexible, since the card is only required to store and transmit the certificate, not to generate it. We therefore consider certificates signed with either ML-DSA or Falcon, the latter offering shorter keys and signatures and thus reducing communication overhead. Evaluating Falcon at the certification level will allow to assess whether adopting distinct signature algorithms for operational use (on-card authentication) and for PKI issuance constitutes a meaningful trade-off in practice. It should be noted, however, that Falcon is not currently approved in the CNSA Suite 2.0 [NSA25], and therefore may not be deployable within the PIV ecosystem as specified today. We nevertheless include it in our evaluation to provide insights applicable to a broader range of authentication frameworks beyond PIV.

4.1 PKI-AUTH and PKI-CAK

Exchanged Data. We first analyze, in Tab. 1, the amount of data exchanged between the system and the card for the PKI-AUTH and PKI-CAK authentication mechanisms. The table allows to compare Protocol 1 (signature-based) and Protocol 4 (KEM-based) in classical and post-quantum settings.

Protocol 1 involves transmitting the card certificate (i.e., the public key and its signature), the challenge issued by the system, and the card's signature over this challenge. In contrast, Protocol 4 requires transmitting the card certificate, the ciphertext produced by the KEM encapsulation, and the MAC used for key confirmation. In Tab. 1, the challenge and the MAC are omitted, as their sizes are negligible compared to the other exchanged data. Moreover, the certificate size accounts only for the public key and its signature; the remaining certificate fields are fixed and therefore not considered here.

Table 1 shows that, although transitioning from classical to post-quantum variants substantially increases the amount of exchanged data, this overhead can be significantly mitigated by adopting Protocol 4 with ML-KEM. At the 128-bit security level, Protocol 1 instantiated with ML-DSA as the post-quantum signature scheme results in 6152 bytes of transmitted data when certificates are signed with ML-DSA, and 4398 bytes when Falcon is used for certificate signatures. In contrast, Protocol 4 instantiated with ML-KEM requires 3988 bytes with ML-DSA-signed certificates and only 2234 bytes with Falcon-signed certificates.

This directly translates into a reduction in communication time. Considering a data rate of 848 kbit/s—the highest standard for contactless communication— Tab. 1 reports in its last column the corresponding communication time, which therefore represents a best-case scenario. For example, with Protocol 1 instantiated with ML-DSA, the communication time is 58 ms when certificates are signed with ML-DSA and 41.5 ms when they are signed with Falcon. With Protocol 4 instantiated with ML-KEM, it drops to 37.6 ms and 21.1 ms, respectively.

In practice, lower data rates would proportionally increase these values. Still, transitioning from the post-quantum Protocol 1 to the ML-KEM-based Protocol 4 yields a relative reduction in communication time of about 35% with ML-DSA-signed certificates and 49% with Falcon-signed certificates.

Overall, in post-quantum settings, these results highlight the benefit of replacing signature-based authentication with a KEM-based approach, as it leads to significantly

Table 1. Size of the transmitted cryptographic material for Protocols 1 and 4, along with the associated communication time. Protocol 1 is instantiated with ECDSA, RSA or ML-DSA for card authentication and ECDSA, RSA, ML-DSA or Falcon for certificate signatures. Protocol 4 is relying on ML-KEM for card authentication and ML-DSA or Falcon for certificate signatures.

Card Algorithm	Certificate Signature	Certificate (bytes)	Signature / Ciphertext (bytes)	Total (bytes)	Comm. @848 kb/s (ms)
Signature-based Protocol (1) — Classical					
ECDSA-256	ECDSA-256	128	64	192	1.8
ECDSA-384	ECDSA-384	192	96	288	2.7
RSA-2048	RSA-2048	512	256	768	7.2
RSA-3072	RSA-3072	768	384	1152	10.9
RSA-4096	RSA-4096	1024	512	1536	14.5
Signature-based Protocol (1) — PQ					
ML-DSA-44	ML-DSA-44	3732	2420	6152	58.0
ML-DSA-65	ML-DSA-65	5261	3309	8570	80.8
ML-DSA-87	ML-DSA-87	7219	4627	11846	111.7
ML-DSA-44	Falcon-512	1978	2420	4398	41.5
ML-DSA-65	Falcon-1024	3232	3309	6541	61.7
ML-DSA-87	Falcon-1024	3872	4627	8499	80.2
KEM-based Protocol (4) — PQ					
ML-KEM-512	ML-DSA-44	3220	768	3988	37.6
ML-KEM-768	ML-DSA-65	4493	1088	5581	52.7
ML-KEM-1024	ML-DSA-87	6195	1568	7763	73.2
ML-KEM-512	Falcon-512	1466	768	2234	21.1
ML-KEM-768	Falcon-1024	2464	1088	3552	33.5
ML-KEM-1024	Falcon-1024	2848	1568	4416	41.7

smaller data exchanges. Moreover, selecting Falcon instead of ML-DSA for signing the card's public key of the certificate can further reduce communication costs, as Falcon produces substantially smaller signatures.

Computational Performances. This section highlights why KEM-based variants are preferable to signature-based ones from a performance perspective, when considering post-quantum algorithms.

A first drawback of ML-DSA lies in its significant execution-time variability, which comes from the rejection sampling inherent in the signing procedure. As an illustration, we evaluate ML-DSA-44 over 10,000 executions in Fig. 1 and report the normalized runtime (relative to the fastest execution), together with its frequency and cumulative distribution. The results reveal a wide dispersion between the shortest and longest runs. Although the average runtime is approximately 2.5× that of the fastest execution, 1% of the runs exhibit execution times between 8.5× and 15× the fastest one.

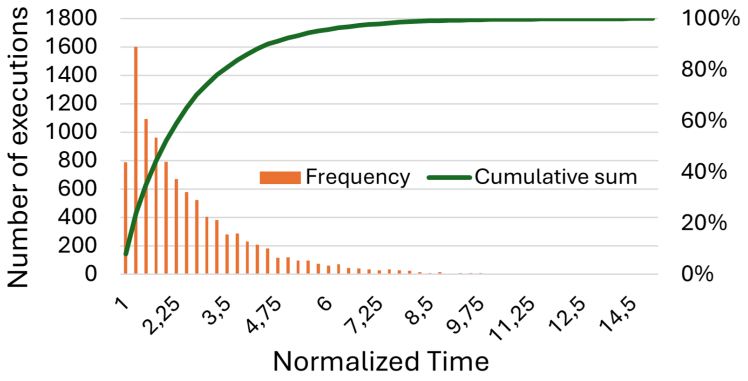


Fig. 1. Distribution of normalized execution times for ML-DSA-44 over 10,000 runs. Execution times are normalized with respect to the fastest observed run and displayed along with their frequency and cumulative distribution, evidencing the pronounced timing variability of the scheme.

A second drawback of ML-DSA is its significantly higher computational cost compared to ML-KEM, even when considering its fastest executions. Table 2 reports the execution times of several algorithms, normalized to the cost of an ECDSA-256 signature.

The timings in Tab. 2 correspond to the main operations executed by the card—namely, step 3 of Protocol 1 and steps 4 and 5 of Protocol 4. We focus exclusively on the card-side operations, as the system does not face the same hardware and computational constraints, making its execution time less critical. Step 3 of Protocol 1 is instantiated with either classical signature schemes (ECDSA or RSA) or the post-quantum signature scheme ML-DSA, while Step 4 of Protocol 4 corresponds to the ML-KEM decapsulation operation.

We observe that ML-DSA-44 costs range from 20.5× to 307.7× that of ECDSA-256, whereas ML-KEM-512 requires only 3.58× the corresponding cost. Consequently, ML-KEM-512 is nearly six times faster than the fastest ML-DSA-44 execution. These results are consistent with performance measurements reported for public implementations, such as those of the pqm4 library (https://github.com/mupq/pqm4), which, al-

though not protected against physical attacks, exhibit a comparable performance gap between ML-DSA and ML-KEM. This contrasts with [FvdHM+23], where the reported ML-KEM implementation appears significantly slower than expected.

Table 2. Normalized execution time of main card computations: step 3 in Protocol 1, and steps 4 and 5 in Protocol 4. All timings are normalized with respect to ECDSA-256. Protocol 1 uses ECDSA, RSA, or ML-DSA, while Protocol 4 relies on ML-KEM and CMAC-AES.

Card Computation	**Normalized Time**
Signature-based Protocol (1) — Classical	**Step 3**
ECDSA-256	1.00
ECDSA-384	1.57
RSA-2048	2.65
RSA-3072	7.08
RSA-4096	15.02
Signature-based Protocol (1) — PQ	**Step 3**
ML-DSA-44	20.5 − 307.7
ML-DSA-65	26.3 − 448.8
ML-DSA-87	36.4 − 518.8
KEM-based Protocol (4) — PQ	**Steps 4 and 5**
ML-KEM-512 + CMAC-AES-128	3.58 + 0.46 = 4.04
ML-KEM-768 + CMAC-AES-192	4.67 + 0.67 = 5.34
ML-KEM-1024 + CMAC-AES-256	6.18 + 0.99 = 7.17

In the KEM-based authentication protocol (Protocol 4), the key-confirmation step requires computing a MAC over the encapsulation ciphertext at Step 5, as described in Sect. 3.1. In our implementation, this MAC is instantiated using CMAC-AES, and Tab. 2 explicitly separates the cost of the key establishment primitive and the associated CMAC. Because ML-KEM ciphertexts are relatively large (ranging from 768 to 1568 bytes depending on the selected security level), the CMAC computation is not negligible in the overall execution cost. In our measurements, for ML-KEM-512 with a 768-byte ciphertext, the CMAC represents $0.46\times$ the cost of ECDSA-256, accounting for approximately 11% of the total KEM+CMAC execution time.

While NIST SP 800-227 [NIS25, Sect. 5.2.2] specifies authenticating the full ciphertext, from a security perspective it is sufficient to authenticate only a truncated

portion of the ciphertext. For instance, computing the MAC over only the first 16 bytes of the ML-KEM-512 ciphertext rather than over the full 768 bytes would reduce the MAC cost from $0.46\times$ to approximately $0.02\times$ the cost of ECDSA-256. The total KEM+CMAC cost would then decrease from $4.04\times$ to about $3.6\times$, making the MAC overhead almost negligible. Such a design trade-off may be attractive in resource-constrained embedded settings, where even moderate reductions in processing time can be meaningful, although a dedicated security analysis would be required to assess the impact of ciphertext truncation on the overall robustness of the protocol.

4.2 SM-AUTH

We now perform the same analysis for the secure messaging protocol SM-AUTH. In particular, we consider the classical version (Protocol 2), instantiated with ECDH and using ECDSA to sign the card certificate, and the post-quantum version (Protocol 5) proposed in Sect. 3.2, instantiated with ML-KEM. As in the authentication case, the certificate signature is instantiated either with ML-DSA or Falcon.

Exchanged Data. In the classical Protocol 2, the exchanged data consist of the card certificate, the system's ECDH public key, the nonce generated by the card, and the computed MAC. In the post-quantum Protocol 5, the exchanged elements are essentially the same, except that the ECDH public key is replaced by the ciphertext resulting from the KEM encapsulation. As before, Tab. 3 omits the nonce and the MAC, and the certificate size includes only the public key and its signature.

Table 3 shows that moving to post-quantum variants significantly increases the amount of exchanged data. However, this overhead remains reasonably contained due to the use of ML-KEM for shared-secret establishment, which offers comparatively moderate data sizes among post-quantum primitives.

Focusing on Protocol 5, and similarly to the authentication case, selecting Falcon rather than ML-DSA for the certificate signature further reduces the communication cost. For instance, at the 128-bit security level, the total exchanged data decreases from 3988 bytes with ML-DSA-signed certificates to 2234 bytes when Falcon is used.

Assuming a throughput of 848 kbit/s, this corresponds to a reduction in communication time from 37.6 ms to 21.1 ms, i.e., an overhead reduction of approximately 44% between the two configurations.

Computational Performances. We now evaluate the computational cost of the secure messaging by comparing the original Protocol 2 with its post-quantum counterpart, Protocol 5. As in the authentication analysis, we focus exclusively on the operations executed by the card. The main differences lie in the key-establishment primitive (ECDH versus ML-KEM) and in the associated CMAC computation. More precisely, we compare lines 7 and 12 of Protocol 2 with lines 8 and 13 of Protocol 5, and restrict our analysis to these operations.

Table 4 reports the normalized execution times of these computations, relative to the cost of ECDH-256 combined with CMAC-AES-128. The table explicitly separates the cost of the key-establishment primitive from that of the CMAC.

Table 3. Size of the transmitted cryptographic material for Protocols 2 and 5, along with the associated communication time. Protocol 2 uses ECDH for key exchange and ECDSA for certificate signatures, while Protocol 5 employs ML-KEM for key exchange and either ML-DSA or Falcon for certificate signatures.

Card Algorithm	Certificate Signature	Certificate (bytes)	Public Key / Ciphertext (bytes)	Total (bytes)	Comm. @848 kb/s (ms)
ECDH-based Protocol (2) — Classical					
ECDH-256	ECDSA-256	128	64	192	1.8
ECDH-384	ECDSA-384	192	96	288	2.7
KEM-based Protocol (5) — PQ					
ML-KEM-512	ML-DSA-44	3220	768	3988	37.6
ML-KEM-768	ML-DSA-65	4493	1088	5581	52.7
ML-KEM-1024	ML-DSA-87	6195	1568	7763	73.2
ML-KEM-512	Falcon-512	1466	768	2234	21.1
ML-KEM-768	Falcon-1024	2464	1088	3552	33.5
ML-KEM-1024	Falcon-1024	2848	1568	4416	41.7

At the 128-bit security level, ML-KEM-512 combined with CMAC-AES-128 requires 3.89× the execution time of ECDH-256 with CMAC-AES-128. The replacement of ECDH by ML-KEM therefore results in a moderate increase in card-side computation.

A non-negligible portion of this overhead originates from the MAC computation. In the post-quantum setting, CMAC is evaluated over the ML-KEM ciphertext (ranging from 768 to 1568 bytes depending on the security level), which is an order of magnitude larger than the ECDH public key processed in the classical setting (64 to 96 bytes). As a result, for ML-KEM-512, the CMAC accounts for 0.44× out of the total 3.89× (about 11%), whereas in the classical setting it represents only about 5% of the execution time. This relative impact becomes even more pronounced at higher security levels, where ciphertext sizes further increase.

As discussed in Sect. 4.1, authenticating the entire ciphertext `ct` may not be strictly necessary from a purely cryptographic standpoint, although it is recommended in NIST SP 800-227 [NIS25, Sect. 4.1.1]. Since the CMAC cost increases with the size of the processed input, restricting the MAC computation to a truncated portion of `ct` could reduce the overhead. For example, in the ML-KEM-512 case, computing the CMAC over only the first 16 bytes of the 768-byte ciphertext would reduce its relative cost from 0.44× to approximately 0.02×, lowering the total KEM+CMAC execution time

Table 4. Normalized execution time of main card computations: steps 7 and 12 in Protocol 2, and steps 8 and 13 in Protocol 5. All timings are normalized with respect to ECDH-256+CMAC-AES-128. Protocol 2 uses ECDH, while Protocol 5 relies on ML-KEM.

Card Computations	**Normalized Time**
Protocol (2) — Classical	**Steps 7 and 12**
ECDH-256 + CMAC-AES-128	0.95 + 0.05 = 1.00
ECDH-384 + CMAC-AES-192	1.56 + 0.07 = 1.63
Protocol (5) — PQ	**Steps 8 and 13**
ML-KEM-512 + CMAC-AES-128	3.45 + 0.44 = 3.89
ML-KEM-768 + CMAC-AES-192	4.49 + 0.64 = 5.13
ML-KEM-1024 + CMAC-AES-256	5.95 + 0.95 = 6.90

from 3.89× to 3.47× the execution time of ECDH-256. Such an approach would also align with the current KDF construction, which already operates on a truncated portion of the public input (i.e., $T_{16}(\texttt{epk}_\texttt{S})$ in Protocol 2 or $T_{16}(\texttt{ct})$ in Protocol 5).

Overall, while the KEM-based secure messaging protocol is computationally heavier than its classical counterpart, the overhead remains manageable.

4.3 A unified KEM-based solution

Interestingly, the transition from the classical ECC-based design to the post-quantum KEM-based approach has a remarkably uniform impact across both authentication and secure messaging protocols. In terms of communication, the total exchanged data increases from roughly 200 bytes in the classical setting (ECDSA/ECDH-256) to approximately 4,000 bytes with ML-KEM-512 and ML-DSA-44 certificates, for both protocol families. Similarly, the card-side execution time increases by roughly a factor of four compared to the ECC-based versions (ECDSA-256 vs. ML-KEM-512 for authentication, and ECDH-256 vs. ML-KEM-512 for secure messaging). This consistent behavior in both communication overhead and execution cost indicates that a unified KEM-based transition leads to a predictable and homogeneous performance impact across the protocol suite.

An additional advantage of adopting a unified KEM-based design for both authentication and secure messaging is code mutualization. Instead of implementing and maintaining both a signature scheme and a key-exchange primitive, the secure element can rely on a single KEM implementation. This reduces code footprint and overall implementation complexity, which is particularly valuable on resource-constrained devices. Beyond code reuse, a unified design also allows credential consolidation: if the same KEM key pair is used for both authentication and secure messaging, the card only needs to store one key pair and its associated certificate, rather than distinct credentials for

each protocol. This reduces memory usage and simplifies provisioning. Naturally, this mutualization is most advantageous in scenarios where the card is not required to embed a post-quantum signature scheme for other functionalities, such as document signing.

Finally, current specifications [NIS22, Sect. 6.2.3.3] already allow the secure messaging key-establishment mechanism to be used for authentication purposes, as the card is authenticated during the establishment of secure messaging. This property remains true in our post-quantum adaptations: the KEM-based secure messaging protocol can likewise serve as an authentication mechanism.

5 Conclusion

This work proposes post-quantum adaptations of the PIV authentication mechanisms PKI-AUTH and PKI-CAK, as well as the secure messaging scheme SM-AUTH. In all cases, we advocate the use of KEM constructions, replacing classical digital signatures (RSA, ECDSA) in PKI-AUTH/CAK and classical key exchange (ECDH) in SM-AUTH. We instantiate these designs on a real embedded platform with implementations hardened against side-channel and fault injection attacks. Using the standardized post-quantum ML-KEM algorithm as the KEM, we provide concrete measurements of execution time and communication costs. Our results show that, for PKI-AUTH/CAK, KEM-based authentication is significantly more efficient than post-quantum signature-based alternatives, both in terms of card-side execution time and volume of exchanged data. For SM-AUTH, replacing ECDH with a post-quantum KEM is natural, resulting in more moderate computational and communication overhead, making the transition practical for resource-constrained secure elements. Overall, the proposed unified KEM-based approach combines efficiency, reduced communication latency, and, when applicable, mutualization of code, keys, and certificates within the secure element, while requiring only limited protocol modifications. Although developed in the context of PIV, the underlying design principles are readily applicable to other authentication and key-establishment protocols facing similar post-quantum migration challenges.

Acknowledgments. The authors would like to thank Vincent Guerin, Jérémy Billaud and Stéphane Schneider for their support with the implementation of the proposed protocols and the performance evaluation. The authors are also grateful to Laurent Grémy for insightful discussions on their security properties.

References

ADH+22. Yawning Angel, Benjamin Dowling, Andreas Hülsing, Peter Schwabe, and Florian Weber. Post quantum Noise. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 97–109, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3548606.3560577.

ANS23. ANSSI. *ANSSI views on the Post-Quantum Cryptography transition*. ANSSI, December 2023. URL: https://cyber.gouv.fr/sites/default/files/document/follow_up_position_paper_on_post_quantum_cryptography.pdf.

BDD22. Luk Bettale, Marco De Oliveira, and Emmanuelle Dottax. Post-quantum protocols for banking applications. In Ileana Buhan and Tobias Schneider, editors, *Smart Card Research and Advanced Applications - 21st International Conference, CARDIS 2022*, volume 13820 of *LNCS*, pages 271–289. Springer, 2022. doi:10.1007/978-3-031-25319-5_14.

BDG25. Luk Bettale, Emmanuelle Dottax, and Laurent Grémy. Post-quantum secure channel protocols for eSIMs: Design, validation and performance analysis. In Sabrina De Capitani di Vimercati and Pierangela Samarati, editors, *Proceedings of the 22nd International Conference on Security and Cryptography, SECRYPT 2025*, pages 108–119. SCITEPRESS, 2025. doi:10.5220/0013507200003979.

BdKM23. Colin Boyd, Bor de Kock, and Lise Millerjord. Modular design of KEM-based authenticated key exchange. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *ACISP 23*, volume 13915 of *LNCS*, pages 553–579. Springer, Cham, July 2023. doi:10.1007/978-3-031-35486-1_24.

BFG+21. Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. Towards post-quantum security for Signal's X3DH handshake. In *Selected Areas in Cryptography*, pages 404–430. Springer International Publishing, 2021.

BSI26. BSI. *Cryptographic Mechanisms: Recommendations and Key Lengths*. BSI, January 2026. URL: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf?__blob=publicationFile&v=14.

BSP+25. Christopher Battarbee, Christoph Striecks, Ludovic Perret, Sebastian Ramacher, and Kevin Verhaeghe. Quantum-safe hybrid key exchanges with KEM-based authentication. Cryptology ePrint Archive, Paper 2025/1228, 2025. URL: https://eprint.iacr.org/2025/1228.

CFHS+21. Sofía Celi, Armando Faz-Hernández, Nick Sullivan, Goutam Tamvada, Luke Valenta, Thom Wiggers, Bas Westerbaan, and Christopher A. Wood. Implementing and measuring KEMTLS, 2021. URL: https://doi.org/10.1007/978-3-030-88238-9_5.

CGL+24. Jean-Sébastien Coron, François Gérard, Tancrède Lepoint, Matthias Trannoy, and Rina Zeitoun. Improved high-order masked generation of masking vector and rejection sampling in Dilithium. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2024(4):335–354, 2024. URL: https://doi.org/10.46586/tches.v2024.i4.335-354, doi:10.46586/TCHES.V2024.I4.335-354.

CGMZ22. Jean-Sébastien Coron, François Gérard, Simon Montoya, and Rina Zeitoun. High-order table-based conversion algorithms and masking lattice-based encryption. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(2):1–40, 2022. URL: https://doi.org/10.46586/tches.v2022.i2.1-40, doi:10.46586/TCHES.V2022.I2.1-40.

CGMZ23. Jean-Sébastien Coron, François Gérard, Simon Montoya, and Rina Zeitoun. High-order polynomial comparison and masking lattice-based encryption. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(1):153–192, 2023. URL: https://doi.org/10.46586/tches.v2023.i1.153-192, doi:10.46586/TCHES.V2023.I1.153-192.

CGTZ23. Jean-Sébastien Coron, François Gérard, Matthias Trannoy, and Rina Zeitoun. Improved gadgets for the high-order masking of Dilithium. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(4):110–145, 2023. URL: https://doi.org/10.46586/tches.v2023.i4.110-145, doi:10.46586/TCHES.V2023.I4.110-145.

DN24. Dominik Dziechciarz and Marcin Niemiec. Efficiency analysis of NIST-standardized post-quantum cryptographic algorithms for digital signatures

in various environments. *Electronics*, 14:70, 12 2024. doi:10.3390/electronics14010070.

DZ26. Emmanuelle Dottax and Rina Zeitoun. Design and performance evaluation of post-quantum authentication for embedded systems: A case study on PIV. In *Proceedings of the 23rd International Conference on Security and Cryptography, SECRYPT*, 2026. To appear.

EFG+22. Thomas Espitau, Pierre-Alain Fouque, François Gérard, Mélissa Rossi, Akira Takahashi, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu. Mitaka: A simpler, parallelizable, maskable variant of Falcon. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022*, pages 222–253, Cham, 2022. Springer International Publishing.

FHK+20. Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-Fourier lattice-based compact signatures over NTRU. https://falcon-sign.info, October 2020.

FvdHM+23. Marc Fischlin, Jonas von der Heyden, Marian Margraf, Frank Morgner, Andreas Wallner, and Holger Bock. Post-quantum security for the extended access control protocol. Cryptology ePrint Archive, Report 2023/352, 2023. URL: https://eprint.iacr.org/2023/352.

HNS+21. Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Fiona Johanna Weber, and Philip R. Zimmermann. Post-quantum Wireguard. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 304–321, 2021. doi:10.1109/SP40001.2021.00030.

HQC25. HQC Team. Hamming quasi-cyclic (HQC). https://pqc-hqc.org/, August 2025.

MPR+24. Dustin Moody, Ray Perlner, Andrew Regenscheid, Angela Robinson, and David Cooper. *Transition to Post-Quantum Cryptography Standards*. NIST, November 2024. Initial public draft. URL: https://doi.org/10.6028/NIST.IR.8547.ipd.

NIS05. NIST. *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, May 2005. NIST SP 800-38B. doi:10.6028/NIST.SP.800-38B.

NIS20. NIST. *Recommendation for Stateful Hash-Based Signature Schemes*, October 2020. NIST SP 800-208. doi:10.6028/NIST.SP.800-208.

NIS22. NIST. *Personal Identity Verification (PIV) of Federal Employees and Contractors*, January 2022. FIPS 201-3. doi:10.6028/NIST.FIPS.201-3.

NIS23. NIST. *Digital Signature Standard (DSS)*, February 2023. NIST FIPS 186-5. doi:10.6028/NIST.FIPS.186-5.

NIS24a. NIST. *Cryptographic Algorithms and Key Sizes for Personal Identity Verification*, July 2024. NIST SP 800-78-5. doi:10.6028/NIST.SP.800-78-5.

NIS24b. NIST. *Interfaces for Personal Identity Verification. Part 1 – PIV Card Application Namespace, Data Model, and Representation*, July 2024. NIST SP 800-73pt1-5. doi:10.6028/NIST.SP.800-73pt1-5.

NIS24c. NIST. *Interfaces for Personal Identity Verification. Part 2 – PIV Card Application Card Command Interface*, July 2024. NIST SP 800-73pt2-5. doi:10.6028/NIST.SP.800-73pt2-5.

NIS24d. NIST. *Interfaces for Personal Identity Verification. Part 3 – PIV Client Application Programming Interface*, July 2024. NIST SP 800-73pt3-5. doi:10.6028/NIST.SP.800-73pt3-5.

NIS24e. NIST. *Module-Lattice-Based Digital Signature Standard*, August 2024. NIST FIPS 204. doi:10.6028/NIST.FIPS.204.

NIS24f. NIST. *Module-Lattice-Based Key-Encapsulation Mechanism Standard*, August 2024. NIST FIPS 203. doi:10.6028/NIST.FIPS.203.

NIS24g. NIST. *Stateless Hash-Based Digital Signature Standard*, August 2024. NIST FIPS 205. doi:10.6028/NIST.FIPS.205.

NIS25. NIST. *Recommendations for Key-Encapsulation Mechanisms*, September 2025. NIST SP 800-227. doi:10.6028/NIST.SP.800-227.

NISnd. NIST. Non-repudiation. https://csrc.nist.gov/glossary/term/non_repudiation, n.d.

NSA25. NSA. Commercial National Security Algorithm Suite 2.0. https://media.defense.gov/2025/May/30/2003728741/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS.PDF, 2025.

Res18. Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018. URL: https://www.rfc-editor.org/info/rfc8446, doi:10.17487/RFC8446.

SSW20. Peter Schwabe, Douglas Stebila, and Thom Wiggers. Post-quantum TLS without handshake signatures. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, page 1461–1480, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3372297.3423350.

SSW21. Peter Schwabe, Douglas Stebila, and Thom Wiggers. More efficient post-quantum KEMTLS with pre-distributed public keys. In *Computer Security – ESORICS 2021*, pages 3–22. Springer International Publishing, 2021.