Improving Crypto-Agility in Operational Technology through Exchangeable Smart Cards

Tobias Frauenschläger and Jürgen Mottok
Laboratory for Safe and Secure Systems (LaS³)
OTH Regensburg
93053 Regensburg, Germany
e-mail: {tobias.frauenschlaeger, juergen.mottok}@oth-regensburg.de

***Abstract*—As industrial and Operational Technology (OT) systems face increasing cryptographic demands, including migration to post-quantum cryptography, the need for crypto-agility has become critical. However, retrofitting constrained embedded devices with new cryptographic capabilities is often impeded by hardware limitations, high certification costs, and operational complexity. In this work, we propose a modular architecture that externalizes cryptographic functionality through exchangeable smart cards. This decouples algorithm support and key storage from the host platform, enabling secure and flexible upgrades. We implement and evaluate this concept using resource-constrained embedded devices and a prototype smart card that supports both traditional and post-quantum algorithms. Our results demonstrate that even full cryptographic offloading is feasible with the constraints of OT environments and that the resulting overhead remains acceptable in typical deployment scenarios. We further analyze the security of the interface between the host and the smart card and outline protection mechanisms based on secure channels suitable for OT deployment.**

***Keywords*-Crypto-Agility; Smart Cards; Operational Technology; Post-Quantum Cryptography; Key Management; Security.**

I. Introduction

The importance of robust security in *Operational Technology* (OT) environments has increased significantly. As industrial systems become increasingly interconnected, they face a growing threat landscape that demands proactive and future-proof security measures [1][2]. At the same time, the impending introduction of *Post-Quantum Cryptography* (PQC) is becoming not only a technical necessity but also a regulatory requirement [3]–[5]. This transition poses considerable challenges for many existing OT systems, particularly those characterized by legacy hardware and limited upgradability.

While software-based cryptographic updates are often technically feasible, they are frequently avoided in practice due to the high cost and complexity of device recertification processes. Furthermore, many OT devices are constrained in terms of processing power and memory capacity. As a result, deploying newer and more computationally demanding cryptographic algorithms is often impractical without significant hardware modifications. Yet, *Crypto-Agility*, the ability to rapidly adopt and switch between cryptographic primitives and protocols, is increasingly considered a significant requirement in security architectures of OT systems [6].

Public-key cryptography and Public-Key Infrastructures (PKIs) are fundamental to many essential security features, such as authentication, encryption, and digital signatures. However, managing cryptographic keys in a secure and scalable way remains a complex task. Bootstrapping trust, securely storing private keys, and maintaining PKIs are non-trivial challenges, particularly when device manufacturers and system operators are distinct entities, with operators often lacking deep cryptographic expertise. Dedicated hardware-based security tokens, such as *Hardware Security Modules* (HSMs), *Secure Elements* or *Trusted Platform Modules* (TPMs), are a proven solution for secure key storage, offering tamper-resistant environments for sensitive operations [7]. However, in Embedded and OT environments, such tokens are typically deployed as soldered chips within a device, inheriting the upgrade issues.

Smart Cards as an exchangeable form of dedicated security tokens have been widely adopted in various domains, such as corporate IT, healthcare, and eGovernment, for user authentication and secure key storage. These devices offer an attractive balance between strong security guarantees and deployment flexibility. By externalizing key storage and key management functions from a host device, smart cards simplify the overall system architecture and reduce the complexity of security-critical software updates. This separation would be particularly valuable in OT contexts, where partial hardware upgrades (e. g., inserting a new smart card) could provide support for modern cryptographic algorithms without requiring intrusive modifications to the host device. Pre-installing cryptographic material, such as keys and certificates, on smart cards further simplifies the bootstrapping process.

To address the challenges of upgrading cryptographic capabilities in OT environments, we propose an architecture based on *exchangeable smart cards*. By decoupling key management and algorithm support from the host device in a modular way, smart cards provide a path toward crypto-agility, enabling gradual migration to modern cryptographic standards without requiring deep changes to existing hardware or firmware, while potentially easing device recertification. The main contributions of this work are:

1) *Algorithm Agnosticism*: Our approach allows the adoption of new cryptographic algorithms without requiring their implementation on the host device, reducing complexity and easing certification.
2) *Flexible Key Deployment*: Our approach enables secure provisioning of trust anchors, certificate chains, private keys, and symmetric pre-shared keys.

3) *Evaluation on Constrained Devices*: We demonstrate and evaluate deployment on resource-constrained hardware.
4) *OT-Specific Security Analysis*: We assess the architecture's security against the unique threats and constraints of OT environments.

The remaining paper is structured as follows. In Section II, related work is discussed. Section III then presents our approach on using exchangeable smart cards for crypto-agility. This is followed by a description of our technical implementation in Section IV and the evaluation on resource-constrained devices in Section V. Furthermore, Section VI presents a security analysis of our approach and proposes possible solutions. Finally, the paper is concluded in Section VII.

II. Related Work

A. Key-Management and Bootstrapping

The secure deployment and renewal of cryptographic credentials in distributed systems, such as OT and IoT, has been a long-standing challenge. Various protocols, such as EST [8], ACME [9], or SCEP [10], have been developed to automate certificate enrollment within PKI infrastructures. Furthermore, the protocol BRSKI [11] and its extension for alternative enrollment protocols BRSKI-AE [12] enable secure bootstrapping of new devices into an existing PKI. Recent research has proposed improvements to make PKI automation more applicable to constrained environments such as OT. These include improved architectural concepts, namely zoned segmentation, decentralized trust anchors, and communication improvements such as more lightweight protocols or compact certificate encodings to reduce overhead in industrial or embedded contexts [13]–[17]. However, the implications of these automation functionalities on crypto-agility are not yet covered explicitly in the literature. In addition, factory-based provisioning techniques using device-unique secrets have been explored to facilitate secure bootstrapping of identity and key material in IoT deployments [18][19].

B. Hardware Offloading of Secrets in OT and IoT

The use of dedicated hardware components to secure long-term secrets is well established. TPMs, smart cards, and HSMs are commonly used to protect private keys against both physical and logical attacks. These technologies are widely deployed in TLS-based systems, including those targeting industrial and IoT applications [20]–[24]. All works consider the improved protection of long-term secrets as the main advantage of such offloading, without elaborating on the influence on crypto-agility.

In particular, the work of Urien has explored the integration of smart cards across various use cases [25]–[30]. These efforts demonstrate the scalability and modularity of smart card-based security tokens when used for identity management and key protection. However, these approaches often rely on highly specialized or proprietary interfaces, leading to significant integration overhead into security libraries. Furthermore, the topic of crypto-agility is not directly addressed in his works.

C. Other Approaches for Crypto-Agility

In [6], crypto-agility challenges and solution approaches specific to OT systems are investigated, with a particular focus on hardware-based approaches. Among the promising solutions for enhancing cryptographic flexibility in OT environments are *Field-Programmable Gate Arrays* (FPGAs) and *SmartNICs*. FPGAs allow in-field reconfiguration of cryptographic functions, enabling updates to algorithms without replacing hardware components. This is especially important for inert systems with long upgrade cycles. Similarly, for some device types, SmartNICs offer a way to offload cryptographic operations from host devices to programmable network interfaces, reducing the load on host systems while enabling support for new algorithms. However, while these technologies offer significant potential for crypto-agility, their applicability in OT is limited by integration complexity and cost.

III. Crypto-Agility Using Smart Cards

In this section, we elaborate on the improvements for crypto-agility by decoupling security functionality from host hardware onto smart cards. First, Subsection III-A outlines how such tokens are integrated into devices. Then, the crypto-agility enhancements are presented in Subsection III-B.

A. Smart Card Integration

The integration setup of a smart card into a host device is depicted in Figure 1. Smart cards can be connected to host systems via various physical interfaces, such as ISO 7816-3 [31] (typically in SIM card ID-000 form factor), I²C or SPI (for circuit-level integration), USB (via external readers and the CCID protocol), or embedded within secure SD cards. Regardless of the interface, communication typically follows the ISO 7816-4 standard [32] using Application Protocol Data Units (APDUs) to send commands and receive their responses.

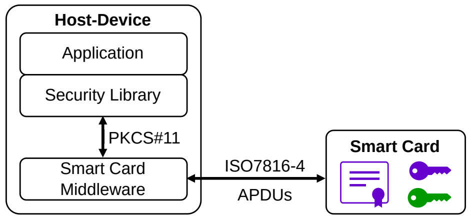

Figure 1. Integration of a smart card into a host device via middleware and standardized interfaces.

To abstract the low-level details of APDU communication, the host device typically uses a middleware layer that interfaces with the smart card and exposes a standardized API to applications. One widely adopted standard is PKCS#11 [33], which defines a common and generic interface for accessing cryptographic tokens. Through this interface, cryptographic artifacts, such as public and private keys, certificates, or symmetric pre-shared keys (PSK), are managed as opaque objects. Operations, e. g., signature generation or data encryption, are invoked through this abstraction and executed on the token without exposing sensitive material to the host.

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

In typical deployments, a security library, such as one that implements TLS or other protocols for secure communications, interacts with the middleware to utilize smart card functionality. The middleware then translates PKCS#11 operations into card-specific APDUs. While manufacturers often provide proprietary middleware specifically for their products, there are also vendor-neutral implementations that support a wide range of smart cards [34].

Smart cards are usually pre-provisioned before their deployment. They may already contain keys, certificates, or other cryptographic material necessary for the device's operation. Importantly, from the perspective of the application, the integration is mostly transparent. After initialization, operations reference specific objects on the card using identifiers or labels, enabling a modular and loosely coupled design.

To ensure secure access to stored cryptographic material, smart cards typically enforce access control mechanisms. The most common protection method is the use of a personal identification number (PIN), which must be presented by the host device to authenticate and authorize access to the card. Only after successful verification of the PIN is the host authorized to perform operations on protected objects, including the use of private keys or the modification of stored certificates. Furthermore, many cards delete their secret data after a specific number of invalid PIN entries. These mechanisms ensure that unauthorized usage is prevented even if the card is stolen. A thorough security discussion is conducted in Section VI.

Finally, the use of pre-personalized smart cards for specific devices also provides a practical solution to the challenge of provisioning device-unique cryptographic secrets during manufacturing. This approach eliminates the need for key generation and injection on the factory floor, streamlining production while maintaining strong security guarantees.

B. Enhancing Crypto-Agility through Smart Cards

The use of smart cards for storing and operating on cryptographic material significantly enhances the cryptographic resilience of OT systems. Long-term artifacts, such as private keys, certificates, or PSKs, can be stored securely on the smart card. These secrets remain non-extractable, and all sensitive operations are executed directly on the card. This architecture protects against unauthorized access or tampering and simplifies the management of cryptographic assets.

Smart cards are well-suited for the operational realities of OT environments. They enable secure provisioning during manufacturing, support lifecycle operations such as credential renewal or revocation, and reduce the operational burden on field devices. Furthermore, their use aligns well with established maintenance workflows and regulatory requirements in the OT domain (e. g., IEC 62351 for the energy grid [35] or IEC 62443 for industrial systems [36], in which hardware-based security is prescribed for specific security levels).

Beyond their role in secure storage, smart cards offer significant advantages in enhancing the crypto-agility of OT systems due to their physical exchangeability. When cryptographic credentials need to be updated (e. g., due to expiry, compromise, or organizational changes), the smart card can be replaced without modifying the host device or its software, assuming the new card provides compatible artifacts with the same identifiers. Even when new identifiers are introduced, only minimal reconfiguration is required.

Support for new cryptographic algorithms can also be added via updated smart cards. In such cases, the host system does not need to implement the new algorithm itself. Instead, it must support the smart card interface to invoke the desired functionality. Typically, this involves updating the host middleware to a version that supports the new APDUs and extends the PKCS#11 interface accordingly. Security libraries interfacing via PKCS#11 usually require only minor adjustments to leverage these extensions due to the generic nature of the API.

This model allows OT devices to adopt new cryptographic standards, such as PQC algorithms, with minimal software changes, streamlining integration and reducing the scope of costly recertification. Since smart cards and their operating systems are often certified as platforms under established security standards (e. g., Common Criteria [37] or FIPS 140 [38]), their integration into existing systems allows manufacturers and operators to reuse these platform certifications within a composite product evaluation of the complete device (host + smart card). As a result, recertifications after modifications to the smart card or the middleware on the host can be much simpler and faster compared to the deployment of the cryptographic functionality solely in software on the host. This significantly reduces both development and compliance overhead, particularly in regulated environments often found in OT. Furthermore, by isolating cryptographic operations from the application logic, smart cards offer a clean separation of concerns, which simplifies security audits and enables clearer security boundaries in system designs. In the context of long-lived OT deployments, this modular approach supports phased upgrades of cryptographic functionality, allowing systems to remain secure and standards-compliant throughout their lifetime. However, each modification of the middleware or the card application interface requires re-evaluation of the composed product, which can limit the extent of certification reuse. In practice, this means that compatible middleware and timely vendor support for new cryptographic features are essential for realizing these benefits.

An alternative to PKCS#11 is the more recently introduced Generic Trust Anchor API (GTA-API) [39]. This API provides a higher-level abstraction between cryptographic applications and the underlying trust anchors. Unlike PKCS#11, which requires applications to be updated with new identifiers or mechanisms when supporting new cryptographic algorithms, the GTA-API offers algorithm-agnostic integration. This enables applications, such as TLS libraries, to transparently benefit from updated smart card capabilities without requiring code changes. This also further decreases the need for recertification due to reduced host-side software changes. While GTA-API holds promise for improving crypto-agility even further, its software ecosystem is still emerging, and integration into production environments remains future work.

IV. Implementation

To demonstrate the practical viability of our proposed crypto-agility enhancement, we implemented a migration scenario from traditional public-key cryptography to PQC on embedded OT devices. This scenario leverages the smart card-based integration architecture described in Section III, allowing cryptographic capabilities to be updated and extended without modifying the host application or firmware. The selected use case involves secure communication over the TLS protocol, which is representative of widely deployed security solutions in industrial and critical infrastructure environments.

Our implementation includes two classes of target devices that reflect the heterogeneity of real-world OT deployments. The first group comprises microprocessor-based platforms running a full Linux operating system. The second group consists of resource-constrained microcontroller-based systems running a real-time operating system (RTOS). In both cases, the devices act as host platforms connected to a smart card, according to the architecture of Subsection III-A.

On each host, a middleware exposes a PKCS#11 interface to the TLS application and handles the low-level APDU communication with the smart card. To support PQC algorithms, our implementation is based on the new version 3.2 of the PKCS#11 standard [33], which introduces identifiers and mechanisms for newly standardized PQC algorithms ML-KEM, ML-DSA, and SLH-DSA. The support in application code for this version was implemented as part of this work.

On the application side, we selected WolfSSL as the TLS library due to its ability to scale across a wide range of hardware, from low-end microcontrollers to high-performance embedded systems, and its established support for PKCS#11 integration. We extended WolfSSL with support for the new version 3.2 interface and incorporated PQC support for the algorithms ML-KEM and ML-DSA. These extensions enable the library to offload sensitive cryptographic operations to the smart card without exposing private key material to the host.

For the middleware layer, we used a prototype implementation provided by Eviden that supports PKCS#11 v3.2 features, including integration with PQC-enabled smart cards. The PQC-capable smart card itself is a prototype developed by Eviden. The interface between this card and the host is based on ISO 7816-3. It supports both traditional cryptographic algorithms (RSA, ECC) and selected PQC algorithms (ML-KEM and ML-DSA). This coexistence of legacy traditional and quantum-safe algorithms on the same platform demonstrates the ability to support a phased migration strategy, a core aspect of crypto-agility in OT deployments.

In addition to support for public-key cryptography, the system also enables the use of symmetric PSKs stored on the smart card. This feature is particularly relevant for use cases requiring an additional layer of security besides public-key cryptography. For TLS integration, the smart card performs key derivation using the PSK object, as required by the TLS 1.3 key schedule. A detailed analysis of PSK integration using external security tokens is presented in [40].

To evaluate the system in practice, we developed lightweight TLS client and server applications that utilize the smart card for various cryptographic operations during the handshake:

- All trusted root certificates are retrieved from the smart card and used as trust anchors for peer authentication.
- A complete certificate chain, consisting of an entity certificate and required intermediate certificates, is read from the card for identity presentation.
- The private key corresponding to the entity certificate is used for handshake signature generation. To ensure that the key remains protected, the TLS handshake transcript (the data to be signed for the handshake signature during authentication) is sent to the smart card, which performs the signing operation internally and returns only the resulting signature.
- If required, the smart card also provides a symmetric PSK for use in the TLS key schedule, computing a session secret without revealing the underlying key material.

In the current implementation, all certificates, both roots and the device chain, are retrieved from the smart card during system initialization and cached in host memory for use during the TLS handshake. As these certificates contain only public data, their potential exposure through a host vulnerability does not present a confidentiality risk. However, storing them in host memory increases the potential attack surface, as compromised or vulnerable host software could tamper with these artifacts. A more secure approach would involve retrieving the certificate chain on demand during the TLS handshake, thereby reducing the window of exposure. Additionally, the signature verification for peer authentication using a root public key should ideally be performed directly on the smart card, preventing the trust anchors from ever being exposed to or manipulated by the host. However, realizing such functionality requires significant modifications to the WolfSSL library and is therefore left for future work.

The host-side software stack has been implemented for two environments: Linux and the embedded RTOS Zephyr. On Linux, the system interfaces with the smart card via USB using the standard CCID protocol. On embedded platforms running Zephyr RTOS, the smart card is accessed directly through the ISO 7816-3 interface. Both implementations are functionally equivalent and demonstrate that the proposed architecture is suitable for a range of hardware classes typically found in OT systems. This hardware-agnostic approach illustrates the effectiveness of smart card-based decoupling, allowing cryptographic upgrades and algorithm changes to be realized without significantly modifying the host software stack. As a result, long-term maintainability is significantly improved.

Finally, custom PKI tooling was developed to support provisioning and bootstrapping tasks. This includes the generation of PQC key pairs and X.509 certificates directly on the smart card prototype, enabling secure device initialization without exposing sensitive material outside the token. In the future, this functionality must be integrated into established PKI systems to use PQC-enabled smart cards in production environments.

V. Evaluation and Feasibility Assessment

To assess the practical feasibility of our smart card-based crypto-agility architecture, we evaluate the performance impact of using an external smart card for cryptographic operations in a typical OT deployment scenario. In line with common OT communication patterns, where long-lived secure connections are typical, we emphasize that connection establishment and initialization occur infrequently. Therefore, our evaluation aims to demonstrate that the overhead introduced by smart card offloading remains acceptable for various typical OT applications. The following subsections detail our measurement setup and present results for handshake time, memory usage, and initialization time, which are critical for constrained embedded systems.

A. Measurement Setup

All measurements are conducted on the microcontroller (MCU) STM32H743ZI from STMicroelectronics (ARM Cortex M7, 480 MHz clock), running Zephyr RTOS version 4.2. Two smart cards from Eviden are used (ID-000 SIM card form factor), connected via an ISO 7816-3 interface directly to the MCU: the commercial CardOS DI V5.3, which is only capable of traditional public-key cryptography, and the prototype of the new CardOS V8 smart card with PQC support mentioned in Section IV. By using both smart cards for the traditional measurements, the possible improvements through upgrading a card in the field while using the same algorithm are shown.

For traditional public-key cryptography, the SECP256R1 elliptic curve is used ("ECC"), while ML-DSA 44 serves as the representative PQC algorithm ("PQC"). The certificate infrastructure consists of a hierarchical PKI with a root certificate authority (CA) and a single intermediate CA that issues device (entity) certificates, resulting in a three-element certificate chain. In all configurations, the TLS key exchange uses traditional ECDHE with the SECP256R1 curve.

B. TLS Handshake Time

TLS handshake time is measured in a mutually authenticated setup, with the MCU acting as the TLS server and the client being a Raspberry Pi 4 running Linux (Raspberry Pi OS Lite, kernel 6.12). The client uses software-only artifacts, while the server is evaluated in three configurations:

1) *Software-only* ("SW-only"): All cryptographic operations are executed in software on the MCU.
2) *Card-signing*: The TLS handshake signature is computed on the smart card; verification of peer certificates is performed on the MCU.
3) *Full offload*: All signature generations and verifications (three per handshake) are delegated to the smart card.

Table I summarizes the measured time-to-first-byte (TTFB) values across all test configurations. Each value represents the average of 100 handshake runs within an isolated network with a round-trip time of about 0.3 ms, measuring the time from the start of the handshake to the receipt of the first application-layer byte on the client side.

TABLE I. TTFB results for ECC and PQC (in milliseconds).

Setup	SW-only	Card-signing	Full offload
ECC (V5.3)	22.92	221.43	3619.67
ECC (V8)		137.78	2961.33
PQC	49.19	453.52	4148.81

The software-only configuration provides a performance baseline for each cryptographic algorithm. As expected, the TTFB increases when cryptographic operations are offloaded to the smart card. The card-signing configuration introduces moderate latency as the handshake transcript must be transmitted to the smart card, and the generated signature is read by the host. The full offload configuration adds a large additional delay, since in addition to signing, all peer signature verifications are delegated to the card. These operations require importing the peer's public keys, verifying the signatures, and removing the imported keys again, adding several round-trips over the card interface in addition to the computations.

The ECC setup consistently shows lower latency than PQC across all configurations. This difference reflects the higher processing requirements and larger data sizes associated with PQC, especially in its current state of maturity. While ECC benefits from decades of optimization and mature hardware-supported implementations, PQC support is still emerging. The current PQC smart card prototype relies on pure software implementations for PQC algorithms. Additionally, PQC artifacts, such as signatures and public keys, are significantly larger, further contributing to transmission and processing delays. Nevertheless, the results demonstrate that even full offload of PQC operations is technically feasible and remains within acceptable bounds for many OT applications, where handshakes occur infrequently and connections are long-lived.

In addition, the reduced ECC measurement results for the newer prototype smart card indicate the potential future improvements possible through an agile system architecture using exchangeable smart cards. Only by upgrading to a newer smart card generation without any software changes (as long as the artifacts on the card use the same identifiers as on the old one), performance can be improved substantially. This also indicates that the currently larger latency of the PQC algorithms will be reduced in the future.

C. Memory Overhead

The peak heap memory usage during the TLS handshake execution is shown in Table II, measured on the MCU platform. For both the commercial smart card and the prototype, the same middleware is used. One of the expected advantages of smart card-based cryptography is the ability to offload computation and reduce memory pressure on the host. However, the current implementation of the middleware has not yet been optimized for resource-constrained embedded deployment. As a result, the measured peak memory usage in the smart card-based configurations is noticeably higher than in the software-only baselines, which are well optimized by WolfSSL for embedded targets.

TABLE II. PEAK HEAP MEMORY USAGE (IN KB).

Configuration	Peak RAM
ECC - Software-only	20.664
ECC - Full offload	36.320
PQC - Software-only	50.832
PQC - Full offload	67.496

The increased memory footprint is primarily attributed to internal buffering, data marshalling, and generic logic within the middleware. These aspects are expected to be significantly reduced through tailored memory management, removal of unnecessary buffers, and streamlined protocol logic. Importantly, the long-term benefits of the smart card-based approach remain valid even with current results and are expected to improve as the implementation matures.

D. Initialization Latency

Finally, Table III shows the initialization latency for both ECC and PQC setups on the MCU platform. This includes middleware startup, smart card initialization, and sequential certificate readout (root, intermediate, and entity certificates).

TABLE III. INITIALIZATION LATENCY (IN MILLISECONDS).

Algorithm	Duration
ECC (V5.3)	999.47
ECC (V8)	859.68
PQC	2920.36

In the SW-only setup, initialization takes less than 1 ms for both algorithms. The difference between the ECC variants again indicates improvements through a newer smart card. The large increase in the PQC setup is mainly due to the significantly larger size of PQC artifacts transmitted through the slow ISO 7816-3 interface. For instance, the PQC entity certificate is around 4182 bytes, compared to just 590 bytes for ECC. Nonetheless, initialization occurs only once per system boot or after a smart card replacement, making it a rare event in OT environments. Given this infrequency, the added latency is acceptable and does not affect ongoing runtime performance.

E. Summary

The evaluation confirms that the use of exchangeable smart cards for cryptographic operations in TLS is feasible on resource-constrained embedded OT devices. While the use of smart cards introduces additional latency during handshake and initialization, these overheads are bounded and acceptable in many OT environments in which secure sessions are long-lived, devices rarely reboot, and safety considerations permit it. The results further demonstrate that even PQC algorithms can be integrated via smart cards without requiring host-side cryptographic implementations. Although the current performance results for PQC are worse than those of ECC due to the unoptimized state of the implementations, future adjustments are expected to improve performance. Overall, the findings validate the practical viability of achieving crypto-agility through smart cards in constrained OT deployments.

VI. SECURITY CONSIDERATIONS

The proposed smart card-based crypto-agility architecture significantly improves modularity and hence longevity in OT systems. However, the use of exchangeable smart cards connected to host devices also introduces new security challenges, particularly when the interface between the host and the smart card is left unprotected. This section analyzes the resulting threat landscape (Subsection VI-A), outlines mitigation strategies (Subsection VI-B), and discusses the achieved security level with extended protections in place (Subsection VI-C).

A. Threat Model

We consider a representative OT deployment in which multiple devices communicate over secure channels, such as TLS. Each device participates in a PKI and contains root certificates, a device-specific certificate chain, and a private key. Optionally, one or more symmetric PSKs may be used. All cryptographic artifacts are stored on a smart card attached to the host device. We assume that the smart card's internal storage is secure, with its tamper resistance being evaluated and certified, and that direct extraction or manipulation of its contents is infeasible.

The smart card serves as the secure execution environment for cryptographic operations, including digital signatures using private keys and symmetric key derivation based on PSKs, and enables peer authentication, either via certificate-based trust anchors or through possession of symmetric PSKs. An attacker's primary goals are to compromise this setup by

- Extracting private keys or PSKs stored on the smart card,
- Modifying trusted root certificates or identity-related data on the card,
- Breaking message integrity or impersonating one peer to enable eavesdropping or man-in-the-middle attacks.

The currently considered setup relies on PIN-based access control: the host device must present a shared PIN to the smart card to enable the retrieval or use of stored cryptographic artifacts for operations such as key exchange, authentication, or signing. However, this mechanism exhibits a critical security flaw. While the PIN may be stored in a secure storage within the host, it is transmitted in plaintext via APDUs during runtime. A local attacker with access to the communication interface between host and card can eavesdrop on the PIN, allowing unauthorized access to the smart card and its stored artifacts. As a result, we identify the following attack scenarios:

1) *Communication Tampering*: An attacker intercepts and modifies the APDU messages between the host and the smart card. This allows the attacker to forge operations, such as generating signatures for malicious data or substituting data read from the card (e. g., certificates).
2) *Smart Card Theft*: If the smart card is physically removed from the host and connected to a malicious device, the attacker can use its credentials to impersonate the original device (host + smart card) in the network.

3) *Malicious Smart Card Insertion*: A manipulated smart card is inserted into the host device. This card may be programmed to leak secrets or respond incorrectly to authentication or signing requests.

These threats highlight the need for a secure association between the host and the smart card, as well as protected communication between them. For our elaborations in the following section, we assume that such a pairing process can be performed within a trusted provisioning environment, enabling the deployment of suitable cryptographic mechanisms.

B. Secure Pairing between Host and Smart Card

To mitigate the threats described above, we propose to establish a cryptographic coupling between the host and the smart card during a secure pairing process. This approach prevents unauthorized hosts or smart cards from being accepted and protects the integrity and confidentiality of all communications between them. Our design is based on two main requirements:

- *Mutual Authentication*: During each session, the host and the smart card must authenticate each other to prevent impersonation or rogue device usage.
- *Secure Channel Establishment*: All APDU exchanges must be cryptographically protected to guarantee message authenticity, integrity, and confidentiality.

To achieve this, the host and the smart card are provisioned with a shared symmetric secret during the secure pairing phase. This secret is then used to derive ephemeral session keys for authenticated encryption of APDU traffic. Furthermore, knowledge of the session keys derived from this secret implicitly authenticates both peers. Public-key cryptography could alternatively be used to establish mutual trust, but managing host-side key material reintroduces the complexity that our smart card-centric design seeks to avoid. Therefore, we prefer symmetric approaches for simplicity and performance.

Two standardized protocols are suitable for this purpose, which are already available in various commercial products:

- *Secure Channel Protocol 03* (SCP03) [41]: Widely adopted in commercial smart cards, SCP03 uses static PSKs to derive session keys for message encryption and authentication. It does not require public-key cryptography and has been formally verified for security [42].
- *Password Authenticated Connection Establishment* (PACE) [43][44]: Originally developed for eIDs and ePassports, PACE maps a user-provided password (or PIN) to a session key using a combination of symmetric and ephemeral public-key cryptography. While more complex and reliant on public-key operations, PACE has also been formally proven secure [45][46]. However, it requires adaptation for use with PQC algorithms [47][48].

Among these, SCP03 provides a lightweight and efficient solution that aligns well with the constraints and goals of OT environments. It supports symmetric authentication, message encryption, and integrity protection without introducing asymmetric key management overhead. In contrast, PACE offers comparable security guarantees, especially if adapted for PQC in the future, but comes with significantly higher complexity due to its use of ephemeral public-key cryptography and more comprehensive protocol steps. While technically feasible, this added complexity makes PACE less attractive for resource-constrained OT systems where simplicity, footprint, and integration effort are critical design considerations.

C. Resulting Protection Level

By integrating a secure communication protocol between the host and the smart card, sensitive APDUs can be cryptographically protected. Protocols like SCP03 and PACE are supported by many commercial smart card platforms, allowing integration into the described architecture with only moderate implementation effort. As a result, communication tampering is impossible, and an attacker is prevented from using a stolen smart card in a malicious device or inserting a malicious smart card, as long as the shared symmetric secret is protected.

Although the secure channel protocols introduce additional processing overhead due to the protection of each exchanged APDU, the impact on overall system performance is expected to be minimal. Since they rely solely on symmetric cryptography, the computational cost is low. Furthermore, the limited speed of the host $\leftrightarrow$ smart card interface further renders the cryptographic overhead less influential compared to the latency of the communication.

Regarding the protection of the shared symmetric secret, the secure internal storage of the smart card is considered secure, ensuring that its stored artifacts remain confidential and tamper-resistant. However, it is assumed that the pairing process itself is performed in a trusted environment. In practice, this assumption may not always hold: while some OT setups allow for secure provisioning before or even after initial deployment, others might require pairing in less controlled field environments. The design of pairing mechanisms that remain secure under such operational constraints represents an important challenge for future research. For the scope of this work, we neglect the detailed pairing process and focus on the security properties once a trusted pairing has been established. In total, overall system security now depends on the protection of the shared symmetric secret on the host.

Modern embedded platforms often include secure storage capabilities within their SoCs. While typically less robust and less rigorously certified than smart card storage, these mechanisms still raise the attacker's cost significantly. Physical access and dedicated setups for side-channel measurements are usually required to extract keys from host memory [49]–[51], and such intrusions are infeasible during normal device operation. In addition, in OT systems, unexpected downtime or physical tampering is likely to trigger rapid alerts or inspections. This further raises the bar for a successful attack.

In summary, the proposed protection scheme eliminates the transmission of PINs in plaintext, ensures mutual trust between the host and the smart card, and protects all exchanged messages. The resulting system achieves crypto-agility, enabling modular upgrades and long-term maintainability in OT environments without expanding the attack surface.

VII. Conclusion and Future Work

This paper introduced a smart card-based architecture to enable crypto-agility in OT systems. By decoupling cryptographic algorithm support and key management from the host device, our approach addresses the challenges of deploying modern cryptography, such as PQC, on constrained and long-lived embedded platforms.

We demonstrated that new cryptographic algorithms can be integrated without requiring host-side implementation, supporting *algorithm agnosticism* and reducing the scope for recertification. Our architecture supports *flexible deployment of cryptographic artifacts*, including both asymmetric key pairs and symmetric keys, using standardized APIs and pre-personalized smart cards. We validated the approach through *practical implementation* on resource-constrained microcontrollers and showed that handshake latency and memory usage remain acceptable in common OT scenarios. Finally, we conducted an *OT-specific security analysis*, addressing the risks of smart card exchangeability and proposing secure pairing mechanisms between hosts and smart cards.

Future work will explore the integration of commercially available smart cards to replace the current prototype and extend support to additional cryptographic algorithms and formats. The implementation of secure communication protocols, such as SCP03 and PACE, together with the creation of the secure pairing process, will further strengthen the protection of sensitive operations between the host and the smart card. Additionally, the emerging GTA-API offers a promising abstraction to simplify application-side integration and will be investigated as a next step toward maximizing crypto-agility.

Acknowledgments

The presented work is part of the research project *KRITIS Scalable Safe and Secure Modules* (KRITIS³M), which is funded by the Project Management Jülich (PtJ) and the German Federal Ministry for Economic Affairs and Climate Action (BMWK) under funding code 03EI6089A.

References

[1] P. Viorel, *Preparing ICS for Future Threats with Quantum-Resistant Cybersecurity*, Nov. 2024. [Online]. Available: https://www.iiot-world.com/ics-security/cybersecurity/preparing-ics-future-threats-quantum-cybersecurity/ (Retrieved: 09/04/2025).

[2] Waterfall team, *How Industrial Cybersecurity Works in 2025*, Jun. 2025. [Online]. Available: https://waterfall-security.com/ot-insights-center/ot-cybersecurity-insights-center/industrial-cyber-security/ (Retrieved: 09/04/2025).

[3] A. Ribeiro, *EU begins coordinated effort for Member States to switch critical infrastructure to quantum-resistant encryption by 2030*, Jun. 2025. [Online]. Available: https://industrialcyber.co/regulation-standards-and-compliance/eu-begins-coordinated-effort-for-member-states-to-switch-critical-infrastructure-to-quantum-resistant-encryption-by-2030/ (Retrieved: 09/04/2025).

[4] National Cyber Security Centre, *Timelines for migration to post-quantum cryptography*. [Online]. Available: https://www.ncsc.gov.uk/guidance/pqc-migration-timelines (Retrieved: 09/04/2025).

[5] Cybersecurity & Infrastructure Security Agency (CISA), *Post-Quantum Considerations for Operational Technology*, Jun. 2025. [Online]. Available: https://www.cisa.gov/resources-tools/resources/post-quantum-considerations-operational-technology (Retrieved: 09/04/2025).

[6] T. Frauenschläger and J. Mottok, "Problems and New Approaches for Crypto-Agility in Operational Technology", in *12th European Congress Embedded Real Time Systems - ERTS 2024*, Jun. 2024. [Online]. Available: https://hal.science/hal-04614197.

[7] M. Khan, M. Ilyas, and O. Bayat, "Enhancing IoT Security Through Hardware Security Modules (HSMs)", in *2024 International Conference on Intelligent Computing, Communication, Networking and Services (ICCNS)*, IEEE, Sep. 2024, pp. 278–282. DOI: 10.1109/iccns62192.2024.10776375.

[8] M. Pritikin, P. E. Yee, and D. Harkins, *Enrollment over Secure Transport*, RFC 7030, Oct. 2013. DOI: 10.17487/RFC7030.

[9] R. Barnes, J. Hoffman-Andrews, D. McCarney, and J. Kasten, *Automatic Certificate Management Environment (ACME)*, RFC 8555, Mar. 2019. DOI: 10.17487/RFC8555.

[10] P. Gutmann, *Simple Certificate Enrolment Protocol*, RFC 8894, Sep. 2020. DOI: 10.17487/RFC8894.

[11] M. Pritikin, M. Richardson, T. Eckert, M. H. Behringer, and K. Watsen, *Bootstrapping Remote Secure Key Infrastructure (BRSKI)*, RFC 8995, May 2021. DOI: 10.17487/RFC8995.

[12] D. von Oheimb, S. Fries, and H. Brockhaus, *BRSKI with Alternative Enrollment (BRSKI-AE)*, RFC 9733, Mar. 2025. DOI: 10.17487/RFC9733.

[13] J. Astorga, M. Barcelo, A. Urbieta, and E. Jacob, "How to Survive Identity Management in the Industry 4.0 Era", *IEEE Access*, vol. 9, 2021. DOI: 10.1109/access.2021.3092203.

[14] J. Höglund, S. Lindemer, M. Furuhed, and S. Raza, "PKI4IoT: Towards public key infrastructure for the Internet of Things", *Computers & Security*, vol. 89, Feb. 2020, Publisher: Elsevier BV. DOI: 10.1016/j.cose.2019.101658.

[15] J. Höglund and S. Raza, "LICE: Lightweight certificate enrollment for IoT using application layer security", in *2021 IEEE Conference on Communications and Network Security (CNS)*, Oct. 2021, pp. 19–28. DOI: 10.1109/CNS53000.2021.9705036.

[16] J. Höglund *et al.*, "AutoPKI: Public key infrastructure for IoT with automated trust transfer", *International Journal of Information Security*, vol. 23, no. 3, pp. 1859–1875, Jun. 2024, Publisher: Springer Science and Business Media LLC. DOI: 10.1007/s10207-024-00825-z.

[17] M. El-Hajj and P. Beune, "Decentralized Zone-Based PKI: A Lightweight Security Framework for IoT Ecosystems", *Information*, vol. 15, no. 6, May 2024, Publisher: MDPI AG, ISSN: 2078-2489. DOI: 10.3390/info15060304.

[18] Q. Zhang, Y. He, Y. Xiao, X. Zhang, and C. Song, "OTA-Key: Over the Air Key Management for Flexible and Reliable IoT Device Provision", *IEEE Transactions on Network and Service Management*, vol. 22, no. 2, Apr. 2025, arXiv:2412.11564 [cs]. DOI: 10.1109/TNSM.2024.3515212.

[19] Fortanix, *Secure Manufacturing of IoT Devices*. [Online]. Available: https://www.fortanix.com/resources/solution-briefs/secure-manufacturing-of-iot-devices (Retrieved: 09/04/2025).

[20] C. Lesjak *et al.*, "Securing smart maintenance services: Hardware-security and TLS for MQTT", in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, 2015, pp. 1243–1250. DOI: 10.1109/INDIN.2015.7281913.

[21] A. J. Paverd and A. P. Martin, "Hardware Security for Device Authentication in the Smart Grid", in *Smart Grid Security*, Springer Berlin Heidelberg, 2013, pp. 72–84, ISBN: 978-3-642-38030-3.

[22] O. Kehret, A. Walz, and A. Sikora, "Integration of Hardware Security Modules into a Deeply Embedded TLS Stack", *In-*

ternational Journal of Computing, vol. 15, pp. 22–30, Mar. 2016. DOI: 10.47839/ijc.15.1.827.

[23] R. Matischek and B. Bara, "Application Study of Hardware-Based Security for Future Industrial IoT", in *2019 22nd Euromicro Conference on Digital System Design (DSD)*, 2019, pp. 246–252. DOI: 10.1109/DSD.2019.00044.

[24] O. Gilles, D. G. Pérez, P. A. Brameret, and V. Lacroix, "Securing IIoT communications using OPC UA PubSub and Trusted Platform Modules", *Journal of Systems Architecture*, vol. 134, p. 102 797, Jan. 2023. DOI: 10.1016/J.SYSARC.2022.102797.

[25] P. Urien, "Innovative TLS 1.3 Identity Module for Trusted IoT Device", in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, 2021, pp. 1–4. DOI: 10.1109/CCNC49032.2021.9369656.

[26] P. Urien, "On Line Secure Elements: Deploying High Security Keystores and Personal HSMs", in *2023 International Conference on Computing, Networking and Communications (ICNC)*, 2023, pp. 450–455. DOI: 10.1109/ICNC57223.2023.10074066.

[27] P. Urien, "Revisiting Multi-Factor Authentication Token Cybersecurity: A TLS Identity Module Use Case", in *2024 International Conference on Computing, Networking and Communications (ICNC)*, 2024, pp. 33–38. DOI: 10.1109/ICNC59896.2024.10556005.

[28] P. Urien, "Innovative Open On-Line Secure Elements Providing Secure Storage and Trusted Computing Resources: Invited Paper", in *2024 Ninth International Conference On Mobile And Secure Services (MobiSecServ)*, vol. CFP24RAC-ART, 2024, pp. 1–6. DOI: 10 . 1109 / MobiSecServ63327 . 2024 . 10759976.

[29] P. Urien, "A New Approach for Crypto Off-loading Based on Personal HSM", in *2023 7th Cyber Security in Networking Conference (CSNet)*, Montreal, QC, Canada: IEEE, Oct. 2023, pp. 23–26. DOI: 10.1109/csnet59123.2023.10339762.

[30] P. Urien, "Personal HSM, Privacy for Subscribers in 5G/6G Networks", in *2022 1st International Conference on 6G Networking (6GNet)*, Paris, France: IEEE, Jul. 2022, pp. 1–6. DOI: 10.1109/6gnet54646.2022.9830453.

[31] ISO/IEC JTC 1/SC 17, "Identification cards — Integrated circuit cards — Part 3: Cards with contacts — Electrical interface and transmission protocols", International Organization for Standardization, Standard ISO/IEC 7816-3:2006, Nov. 2006.

[32] ISO/IEC JTC 1/SC 17, "Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange", International Organization for Standardization, Standard ISO/IEC 7816-4:2020, May 2020.

[33] D. Bong and G. Scott, *PKCS #11 Specification Version 3.2*, OASIS Standard, Apr. 2025. [Online]. Available: https://docs.oasis - open . org / pkcs11 / pkcs11 - spec / v3 . 2 / pkcs11 - spec - v3.2.html.

[34] OpenSC team, *OpenSC*, Jul. 2025. [Online]. Available: https://github.com/OpenSC/OpenSC (Retrieved: 09/04/2025).

[35] International Electrotechnical Commission, "Power systems management and associated information exchange – Data and communications security", Standard IEC/TS 62351:2025, 2025.

[36] International Electrotechnical Commission, "Industrial communication networks – Network and system security", Standard IEC/TS 62443:2009, 2009.

[37] ISO/IEC JTC 1/SC 27, "Information technology — Security techniques — Evaluation criteria for IT security", International Organization for Standardization, Standard ISO/IEC 15408-1/2/3:2020, Dec. 2020.

[38] National Institute of Standards and Technology (US), "Security requirements for cryptographic modules", National Institute of Standards and Technology, Washington, D.C., Tech. Rep., 2019. DOI: 10.6028/nist.fips.140-3.

[39] ISO/IEC JTC 1/SC 41, "Internet of Things (IoT) — Generic trust anchor application programming interface for industrial IoT devices", International Organization for Standardization, Standard ISO/IEC TS 30168:2024, May 2024.

[40] T. Frauenschläger, L. Füreder, and J. Mottok, "Enhancing Quantum-Safe Cryptography in TLS: The Role of Pre-Shared Keys", in *2025 International Conference on Applied Electronics (AE)*, IEEE, Sep. 2025.

[41] GlobalPlatform Technology, *Secure Channel Protocol 03*, Apr. 2020. [Online]. Available: https://globalplatform.org/specs-library/secure-channel-protocol-03-amendment-d-v1-2.

[42] M. Sabt and J. Traoré, "Cryptanalysis of GlobalPlatform Secure Channel Protocols", in *Security Standardisation Research. Lecture Notes in Computer Science*, Springer International Publishing, 2016, pp. 62–91. DOI: 10.1007/978-3-319-49100-4_3.

[43] Federal Office for Information Security, *BSI TR-03110: Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token*, 2016. [Online]. Available: https: / / www . bsi . bund . de / EN / Themen / Unternehmen - und - Organisationen / Standards - und - Zertifizierung / Technische - Richtlinien/TR- nach- Thema- sortiert/tr03110/tr- 03110.html?nn=909310 (Retrieved: 09/04/2025).

[44] ICAO, *Doc 9303: Machine Readable Travel Documents*, 2021. [Online]. Available: https://www2023.icao.int/publications/Documents/9303_p11_cons_en.pdf (Retrieved: 09/04/2025).

[45] J. Bender, M. Fischlin, and D. Kügler, "Security Analysis of the PACE Key-Agreement Protocol", in *Information Security. ISC 2009. Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2009, pp. 33–48. DOI: 10.1007/978-3-642-04474-8_3.

[46] J.-S. Coron, A. Gouget, T. Icart, and P. Paillier, "Supplemental Access Control (PACE v2): Security Analysis of PACE Integrated Mapping", in *Cryptography and Security: From Theory to Applications. Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2012, pp. 207–232. DOI: 10.1007/978-3-642-28368-0_15.

[47] N. Alnahawi, J. Alperin-Sheriff, D. Apon, G. T. Davies, and A. Wiesmaier, *NICE-PAKE: On the Security of KEM-Based PAKE Constructions without Ideal Ciphers*, Cryptology ePrint Archive, Paper 2024/1957, Publication info: Preprint., 2024. [Online]. Available: https : / / eprint . iacr . org / 2024 / 1957 (Retrieved: 07/15/2025).

[48] N. Alnahawi *et al.*, *Post-Quantum Cryptography in eMRTDs: Evaluating PAKE and PKI for Travel Documents*, Cryptology ePrint Archive, Paper 2025/812, Publication info: Preprint., 2025. [Online]. Available: https://eprint.iacr.org/2025/812 (Retrieved: 07/15/2025).

[49] T. Krachenfels, T. Kiyan, S. Tajik, and J.-P. Seifert, "Automatic extraction of secrets from the transistor jungle using Laser-Assisted Side-Channel attacks", in *30th USENIX Security Symposium (USENIX Security 21)*, USENIX Association, Aug. 2021, pp. 627–644, ISBN: 978-1-939133-24-3. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/krachenfels.

[50] K. Murdock *et al.*, "Plundervolt: Software-based Fault Injection Attacks against Intel SGX", in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 1466–1482. DOI: 10.1109/SP40000.2020.00057.

[51] H. Lohrke, S. Tajik, T. Krachenfels, C. Boit, and J.-P. Seifert, "Key Extraction Using Thermal Laser Stimulation: A Case Study on Xilinx Ultrascale FPGAs", *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 573–595, Aug. 2018. DOI: 10.46586/tches.v2018.i3.573-595.