Enhancing LoRaWAN Security: Addressing Static Root Keys with Post-Quantum Cryptography

Matheus de O. Saldanha[1], **Alexandre A. Giron**[2], **Ricardo Custódio**[1], **Thaís B. Idalino**[1]

[1]Universidade Federal de Santa Catarina (UFSC)
Florianópolis - SC - Brazil

[2]Universidade Tecnológica Federal do Paraná (UTFPR)
Toledo - PR - Brazil

matheus.saldanha@posgrad.ufsc.br, alexandregiron@utfpr.edu.br,

{ricardo.custodio,thais.bardini}@ufsc.br

Abstract. *The lack of an update method for LoRaWAN's static root keys poses a major security risk, exposing networks to key compromise. Furthermore, existing approaches that use public-key cryptography are vulnerable to emerging quantum threats. This paper presents a quantum-safe solution to dynamically update root keys through a Root Key Renewal procedure, using post-quantum authenticated Diffie–Hellman via CSIDH and post-quantum signatures. To avoid a similar stagnation of the new signature key pairs, a Key Pair Update procedure is also proposed to refresh them. The proposed solution is tested in a simulated setup and the results show that HAWK, SNOVA, MAYO, Falcon, and SQISign are viable candidates for post-quantum LoRaWAN deployments.*

1. Introduction

The Internet of Things (IoT) is revolutionizing industries by enabling smart automation across various domains [Issac et al. 2020]. Low-Power Wide-Area Networks (LPWANs) play a crucial role in this transformation, providing long-range, low-power communication for constrained devices [Mekki et al. 2018]. Among LPWAN technologies, LoRaWAN is a leading standard known for its reliability and security [Almuhaya et al. 2022]. However, it remains vulnerable to attacks such as eavesdropping, replay attacks, and key compromise [Ntshabele et al. 2022, Butun et al. 2018].

A major security risk in LoRaWAN is the static root key issue [Butun et al. 2018]. In the Over-The-Air Activation (OTAA) process, the End Device (ED) and the Join Server (JS) derive session keys from preconfigured root keys, which remain static and unchanged throughout the entire lifetime of the device [LoRa Alliance Technical Committee 2017]. If compromised, these AES-128 root keys provide attackers with persistent access, endangering the entire network. Although previous solutions have explored root key updates using symmetric cryptography [Hayati et al. 2022, Tsai et al. 2022], secure key exchange protocols [Qadir et al. 2023, Milani and Chatzigiannakis 2021], and external key management strategies [Donmez and Nigussie 2019], they do not address the growing threat of quantum computing.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001

Quantum computers pose a serious threat to classical cryptographic schemes such as RSA and ECDSA, which can be efficiently broken using Shor's algorithm [Shor 1997]. In response, post-quantum cryptography (PQC) has emerged, grounded in hard mathematical problems believed to be resistant to quantum attacks. As part of its first PQC standardization process, NIST recently selected several algorithms for standardization, including ML-DSA, SLH-DSA, and Falcon for digital signatures, and ML-KEM and HQC for key encapsulation mechanisms (KEMs) [Alagic 2025], while additional digital signature algorithms remain under evaluation in a new process [Alagic et al. 2024].

Despite this progress, finding a quantum-resistant alternative to Diffie-Hellman remains an open challenge. The most promising candidates lie within the realm of isogeny-based cryptography. One such approach, SIKE — the KEM variant of SIDH [Jao and De Feo 2011] — initially showed promise and was included in the first NIST PQC process before being broken [Robert 2023]. A more optimistic candidate is CSIDH [Castryck et al. 2018], which leverages group actions on supersingular elliptic curves. CSIDH provides strong compactness in public key size, positioning it as a viable post-quantum successor to ECDH.

This paper presents the first PQC-based approach to mitigate the static root key vulnerability in LoRaWAN. Although [Figlarz and Hessel 2024] explores PQC in LoRaWAN, it does not address this specific issue. A post-quantum authenticated Diffie-Hellman key exchange is proposed to enable dynamic root key updates between the ED and the JS. To accomplish this, CSIDH and NIST post-quantum digital signatures — including the additional candidates — are integrated into a Root Key Renewal (RKR) procedure, periodically updating the root keys. In addition, to efficiently derive the root keys from the CSIDH shared secret, the lightweight Ascon-Hash256 hash function from the Ascon family of symmetric-key primitives is used, NIST's standard for lightweight cryptography [Dobraunig et al. 2021, Sonmez Turan et al. 2023].

With the introduction of the signature key pair, the longevity of the new key pair must be addressed; otherwise, the problem is merely shifted. If the signing keys remain static, they become the next single point of failure. To prevent this, a complementary Key Pair Update (KPU) procedure is introduced to periodically refresh the signature key pair, supported by the use of digital certificates. The performance evaluation of PQC algorithms within these procedures demonstrates the feasibility of integrating select candidates into the solution. The overhead is further minimized by the infrequent nature of RKR, and even more so in the case of KPU.

In summary, the main contributions of this paper are as follows:

(i) Proposal of post-quantum RKR and KPU procedures to securely update root and signature keys;
(ii) Evaluation of execution time and energy consumption of the selected post-quantum algorithms in an embedded setting;
(iii) Analysis of Time-on-Air (ToA) for LoRaWAN messages, and overall procedures completion time and energy consumption.

The remainder of this paper is organized as follows. Section 2 presents the necessary background on the core components of the proposed solution. Section 3 reviews related work in the area. In Section 4, the solution to the static root key vulnerability is in-

troduced. Section 5 describes the performance evaluation and discusses the main results. Finally, Section 6 concludes the paper and outlines future work.

2. Background

This section provides background on LoRaWAN, with emphasis on its root key security limitations. The CSIDH post-quantum key exchange scheme is then presented, as it serves as the core mechanism for updating root keys between the End Device (ED) and the Join Server (JS). In addition, post-quantum digital signatures and the Ascon hash function are introduced to enable a lightweight and authenticated key exchange between the parties.

2.1. LoRaWAN

LoRaWAN, standardized by the LoRa Alliance, is a low-power, long-range IoT protocol that uses Semtech's LoRa modulation with chirp spread spectrum (CSS) for interference resistance. Operating in the unlicensed sub-GHz band, it optimizes transmission by limiting packet sizes to reduce ToA — the elapsed transmission time of a packet between the ED and the gateway (GW) — and allowing configurable parameters such as bandwidth (BW), coding rate (CR), and spreading factor (SF) to balance range, robustness, and data rate (DR) [Almuhaya et al. 2022]. For instance, in the EU863-870 band, the DRs range from 250 bps (DR0) to 11 kbps (DR6), with a maximum payload of 222 bytes [LoRa Alliance Technical Committee Regional Parameters Workgroup 2022].

LoRaWAN follows a star-of-stars topology where EDs communicate with GWs via LoRa modulation. The GWs forward messages to the Network Server (NS), which handles authentication, routing, and mobility. The NS also interfaces with the Application Server (AS) for data processing and user services, and with the Join Server (JS) for authentication and key distribution [Ribeiro et al. 2019].

LoRaWAN security relies on two root keys: AppKey and NwkKey, which establish secure connections between the ED and the AS and NS, respectively. These root keys serve as the foundation for deriving session keys that ensure data confidentiality and authenticity using AES encryption and AES-CMAC, correspondingly. To mitigate the risk of key compromise, the LoRaWAN specification [LoRa Alliance Technical Committee 2017] recommends storing these static root keys in secure elements (SEs) or hardware security modules (HSMs).

Devices join the network via Activation by Personalization (ABP) or Over-The-Air Activation (OTAA). ABP preconfigures session and root keys for immediate communication, but prevents key updates, reducing security. OTAA enhances security by storing only root keys in the ED, dynamically deriving session keys through the Join Procedure with the JS. While OTAA allows session key updates via the Rejoin Procedure, it lacks a root key renewal mechanism. This poses a security risk — if root keys are compromised, the entire network is vulnerable [Butun et al. 2018].

2.2. CSIDH

CSIDH [Castryck et al. 2018] is a post-quantum key exchange protocol that serves as an isogeny-based alternative to ECDH. Instead of relying on number-theoretic and elliptic curve point multiplication problems, CSIDH is built on the hardness assumption of finding

isogenies between supersingular elliptic curves, which are rational maps that map one elliptic curve to another while preserving much of their structure.

The primitive supports public key sizes of 64, 128, and 224 bytes, with private keys being half that size when optimally encoded. In terms of security, it provides 128-bit classical security and a conjectured 64-bit post-quantum security when using 64 byte keys. The original paper reports an estimated key exchange time of 80 ms, demonstrating its practicality [Castryck et al. 2018]. This suggests that CSIDH is best suited for applications where bandwidth constraints are a greater concern than computational efficiency.

2.3. Post-Quantum Digital Signatures

Classical digital signatures, such as RSA and ECDSA, rely on mathematical problems that quantum computers can efficiently solve using Shor's algorithm [Shor 1997], making them obsolete in a post-quantum world. To address this, PQC introduces digital signatures based on quantum-resistant assumptions.

The NIST PQC standardization process has selected ML-DSA, SLH-DSA, and Falcon as the first post-quantum signature standards [Alagic 2025]. A new process [Alagic et al. 2024], currently in Round 2, is now evaluating 14 additional candidates across various cryptographic families, including multivariate (UOV, MAYO, SNOVA), lattice-based (HAWK), code-based (LESS, CROSS), isogeny-based (SQISign), MPC-in-the-Head (Mirath, MQOM, PERK), and symmetric-based (FAEST) approaches.

2.4. Ascon Hash Function

The Ascon family of symmetric-key cryptographic primitives [Dobraunig et al. 2021] was selected in 2023 as NIST's standard for lightweight cryptography. It encompasses Authenticated Encryption with Associated Data (AEAD), hash function, and Extendable Output Functions (XOFs) [Sonmez Turan et al. 2023], all designed to provide strong security, high efficiency, and flexible implementation. These characteristics make Ascon particularly suitable for resource-constrained environments, including IoT devices, embedded systems, and low-power sensors.

This work focuses on Ascon-Hash256, a permutation-based hash function optimized for lightweight cryptographic use cases that generates a 256-bit digest.

3. Related Works

The static nature of LoRaWAN's root keys presents a critical security vulnerability, exposing networks to long-term risks if the keys are compromised [Butun et al. 2018, Ntshabele et al. 2022]. To address this, previous work has explored symmetric-key approaches for key refresh [Chen et al. 2021, Papatsaroucha et al. 2024], public-key methods for secure exchanges [You et al. 2018, Marlind and Butun 2020], and external key management solutions to enhance overall security [Ribeiro et al. 2019].

The *Assisted Mode* introduced in [Donmez and Nigussie 2019] delegates key management to a master device positioned between the Join Server (JS) and the End Device (ED). It establishes *supersessions* to enable regular root key updates, emphasizing low-cost implementation and forward secrecy, mitigating security risks associated with device compromise. The study is motivated by a medical use case, where regular key

updates are vital to ensuring security. However, its reliance on close physical proximity between devices restricts scalability.

By employing symmetric cryptographic techniques, [Hayati et al. 2022] introduces a method to periodically refresh root keys using CTR_AES DRBG 128. This approach involves new Join/Rejoin message exchanges at previously synchronized intervals to securely transmit updated keys from the JS to the ED. The work includes formal verification, along with additional validation procedures, such as a randomness test, ensuring the security of the approach. However, it lacks backward compatibility, which may hinder integration with existing deployments. Along similar lines, [Tsai et al. 2022] presents the Two-Stage High-Efficiency Encryption Key Update Scheme (THUS), which updates both root and session keys via the Root Key Update (RKU) and Session Key Update (SKU) processes. The scheme replaces the AES S-Box with a dynamic S-Box (D-Box) to improve security and efficiency. The work also includes key features, such as mutual authentication and confidentiality, although real-world performance testing is still needed.

In [Qadir et al. 2023], a Key Generation and Distribution (KGD) mechanism is proposed to enhance the LoRaWAN Join Procedure using public-key cryptography. This approach uses Argon2 to seed a deterministic random bit generator (DRBG) for root key generation and employs ECDH and ECDSA for secure key exchanges. While the design improves key management, its overall performance still requires further evaluation. Similarly, [Milani and Chatzigiannakis 2021] addresses the root key update challenge using elliptic curve cryptography (ECC), specifically through ECDH to establish shared secrets. This method prioritizes forward and backward secrecy, as well as computational and decisional secrecy. The work also tests the solution across various devices and concludes that overhead is acceptable given the low frequency of key updates.

To the best of our knowledge, [Figlarz and Hessel 2024] is the only work that introduces PQC in LoRaWAN to protect against quantum threats. However, it does not address the vulnerability of static root keys. The authors propose replacing the AES-128 AppKey with the Kyber KEM-1024 algorithm to enhance quantum resistance. Despite this, the solution remains susceptible to long-term compromise, as no root key refresh mechanism is provided.

In contrast to prior approaches, this work enables secure root key refreshment through a quantum-safe Root Key Renewal (RKR) procedure. By combining CSIDH with NIST PQC digital signatures, it establishes a post-quantum authenticated Diffie-Hellman to derive new root keys. To avoid similar long-term exposure of the introduced signature key pair, a complementary Key Pair Update (KPU) procedure ensures periodic refresh. The performance analysis demonstrates the practicality of the proposed solution for quantum-safe LoRaWAN deployments.

4. Proposed Solution

To mitigate the static root key vulnerability in LoRaWAN, a novel Root Key Renewal (RKR) procedure is proposed in which the End Device (ED) and the Join Server (JS) establish a shared secret prior to the Join Procedure, as shown in the upper section of Figure 1. This shared secret is derived using CSIDH and serves as input to the lightweight hash function Ascon-Hash256, which deterministically produces new root keys: AppKey and NwkKey, as illustrated in the lower section of Figure 1.

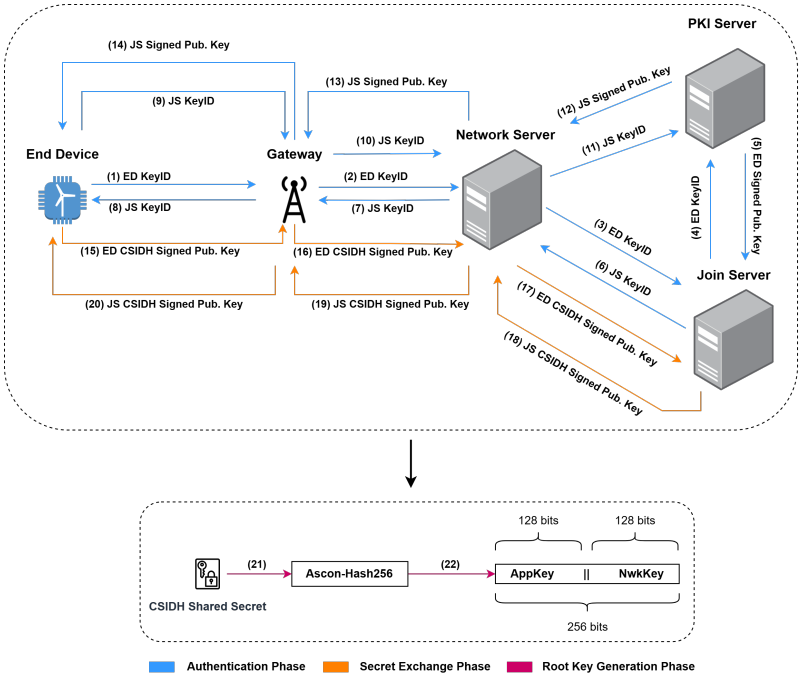

Figure 1. Root Key Renewal and Key Update Procedures

To ensure authenticity, the exchanged CSIDH public keys are signed using post-quantum digital signatures, requiring both the ED and JS to possess digital signature key pairs. Typically, mutual authentication would involve exchanging certificates containing public keys. However, this is impractical in LoRaWAN due to bandwidth limitations. To address this, a public key infrastructure (PKI) server is introduced to manage and store digital certificates — itself also having a signature key pair trusted by both the ED and JS. These certificates bind a public key to an entity identifier — DevEUI for EDs and JoinEUI for JSs [LoRa Alliance Technical Committee 2017] — and are indexed in the database by a compact KeyID, defined as the last 48 bits of the Ascon-Hash256 digest of the public key. These KeyIDs are small enough to be transmitted in a single LoRaWAN packet across all EU863-870 Data Rates (DRs), which is adopted as the reference scenario. While this introduces some communication and management overhead, the design shifts the storage burden to the PKI server, which manages certificates at scale. Meanwhile, the ED performs cryptographic operations without the need to store multiple certificates.

Communication with the PKI server differs for each party: the JS uses a standard TCP/IP link, as it does with the Application Server (AS) and Network Server (NS), while the ED communicates over the LoRa link, routed through the Gateway (GW) and NS, similar to its communication with the AS and JS. Due to the large size of public keys and signatures, the exchanged data typically needs to be segmented into multiple packets, introducing communication overhead. However, certain post-quantum signature schemes offer compact representations that, in some cases, approach the limits of fitting within a single packet, thereby reducing this impact, as demonstrated in Section 5. This mechanism can also be periodically repeated to refresh root keys in advance of the Rejoin Procedure.

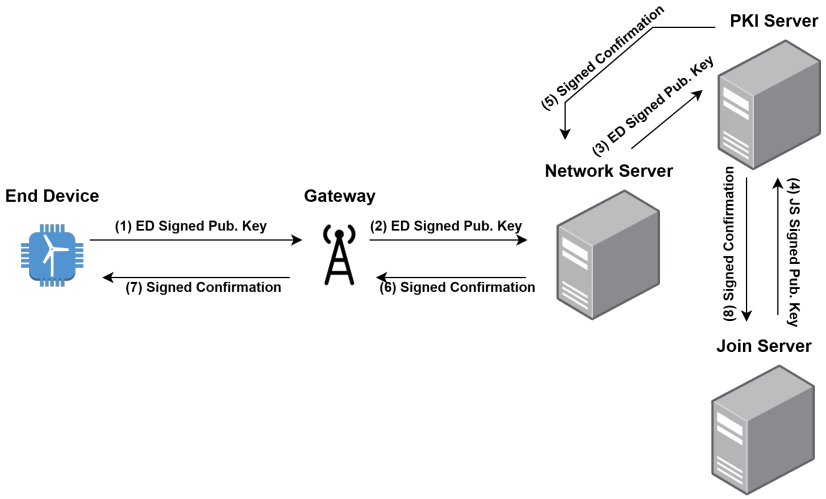

Figure 2. Key Pair Update Procedure

However, while replacing static root keys improves security, failing to update the introduced signature key pairs would merely shift the static key problem to a new layer. To address this, a complementary KPU procedure is introduced, depicted in Figure 2. This procedure enables both ED and JS to periodically generate fresh digital signature key pairs. During this process, the newly generated public key is then sent to the PKI server, signed with the corresponding old private key. Upon successful verification, the PKI server issues a new certificate, updates its database, and sends a confirmation back to the requesting party, without introducing significant communication overhead.

If any verification fails during a procedure, the respective party silently aborts without sending a response, similar to the behavior of the LoRaWAN Join Procedure [LoRa Alliance Technical Committee 2017].

4.1. Root Key Renewal Procedure

The RKR procedure is shown in Figure 1 and it consists of three phases: authentication, secret exchange, and root key generation phase, as described next.

4.1.1. Authentication Phase

The authentication process begins with messages (1) through (3), where the ED sends its KeyID to the JS. The JS queries the PKI server, verifying the certificate and retrieving the corresponding public key, as shown in messages (4) and (5), verifying the PKI server's signature. Then, the JS sends its own KeyID to the ED, which follows an analogous procedure, as shown in messages (6) through (14). Once both parties successfully verify each other's identities, the authentication phase is complete.

4.1.2. Secret Exchange Phase

In the secret exchange phase, the ED sends its CSIDH public key, signed with its post-quantum digital signature private key, to the JS, as shown in messages (15) to (17). The

JS verifies the signature against the previously received public key from the PKI server and responds with its own signed CSIDH public key in messages (18) through (20). The ED performs the same verification to ensure authenticity. Once both CSIDH public keys are exchanged, the parties complete the CSIDH protocol to derive a shared secret.

4.1.3. Root Key Generation Phase

In the root key generation phase, both the ED and JS independently generate the new root keys locally, using the shared secret obtained in the previous phase as input to the Ascon-Hash256 cryptographic hash function, as shown in step (21). The hash function outputs a 256 bit value, where the first half becomes the AppKey and the second one the NwkKey, as illustrated in step (22). These keys serve as the foundation for secure communication in LoRaWAN. With the new root keys established, the Join/Rejoin Procedure can be executed to generate updated session keys and delegate them correspondingly to the AS and NS, ensuring continued security.

4.2. Key Pair Update Procedure

In addition to establishing root keys securely, a mechanism is introduced to periodically update the post-quantum digital signature key pairs, the KPU procedure, thereby preventing key stagnation, as illustrated in Figure 2. When either the ED or the JS initiates a key update, it locally generates a new key pair and sends a certificate update request to the PKI server. This request includes the new public key signed with the corresponding old private key, ensuring the authenticity of the update.

The certificate update request containing the signed public key is transmitted through messages (1) to (3) for the ED and in message (4) for the JS. Upon verifying the signature, the PKI server issues a new certificate and updates its internal database. A single byte signed confirmation message is then sent back to the requesting party, as indicated in messages (5) through (7) for the ED and in message (8) for the JS, each verifying the received signature.

5. Performance Evaluation and Discussion

To assess the feasibility of integrating post-quantum cryptographic algorithms — specifically CSIDH-512 and various post-quantum digital signature schemes — within a LoRaWAN environment, a performance evaluation is carried out from the perspective of the End Device (ED) in a simulated setting. The evaluation considers both the Root Key Renewal (RKR) and Key Pair Update (KPU) procedures. It begins by measuring the execution time and energy consumption of each cryptographic function on a Raspberry Pi Zero W, selected to approximate the constraints of a typical LoRaWAN ED. Then the Time-on-Air (ToA) for each transmitted message is analyzed. Finally, the total energy cost of each procedure is calculated by summing the energy required for local computations, while the overall runtime is derived by combining both computation and communication durations.

Given the resource constraints of LoRaWAN devices, the evaluation targets the most lightweight variant of each algorithm that meets the NIST Level 1 security requirements from Round 2 submissions[1]. The selected variants are: ML-DSA-44, SLH-DSA

[1]https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures . RYDE, Mirath and SDitH are

Table 1. Performance Metrics for CSIDH

Function	Energy Consumption (J)	Std. Dev. Energy Cons. (J)	Time (ms)	Std. Dev. Time (ms)
Secret Key Generation	0.0000177156	0.000003215	0.038	0.063
Public Key Generation	4.576054476	0.128672	5889.388	372.912
Shared Secret Generation	4.528570452	0.134987	5828.276	446.146

Table 2. Public Key and Signature Sizes in Bytes for Post-Quantum Algorithms

Algorithm	**ML-DSA**	**SLH-DSA**	**Falcon**	**MiRitH**	**MQOM**	**PERK**	**MIRA**
Public Key	1312	32	897	129	80	150	84
Signature	2420	17088	666	7877	4164	8360	7376
Algorithm	**SDitH**	**MAYO**	**QR-UOV**	**SNOVA**	**UOV**	**LESS**	**CROSS**
Public Key	132	1420	24271	1016	278432	41788	54
Signature	10684	454	200	248	128	1329	12472
Algorithm	**SQISign**	**HAWK**	**FAEST**				
Public Key	65	1024	32				
Signature	148	555	5060				

Haraka Simple 128f, Falcon 512, MiRitH Ia fast, MQOM2-L1-gf256-fast-3r, PERK I-fast3, MIRA 128f, SDitH gf251-L1-thr, MAYO-1, QR-UOV I-(127, 156, 54, 3), SNOVA (24, 5, 4), UOV Ip-classic, LESS-252-68 1, CROSS R-SDP(G) 1 fast, SQISign I, HAWK 512, and FAEST EM-128f. Table 2 summarizes their respective public key and signature sizes according to the submissions. Additionally, to support 32-bit architectures, the original CSIDH implementation[2] is adapted by eliminating the use of 128-bit integers, which are generally not supported on such platforms. In particular, 64-bit integer multiplication — previously relying on 128-bit intermediates — is now handled by a custom routine. This method splits each 64-bit operand into two 32-bit halves, performs partial multiplications, and then combines the results, avoiding the need for native 128-bit types. Detailed performance metrics, including standard deviations, along with the source code, are available in a public repository[3].

5.1. Time and Energy Consumption

As previously noted, the evaluation of execution time and energy consumption for each algorithm function is conducted on a Raspberry Pi Zero W, which features a 1GHz BCM2835 single-core ARMv6 processor, 512MB RAM, and a 32-bit microarchitecture. Execution times are averaged over 10,000 iterations to ensure consistency, while energy consumption is measured across 100 iterations using a Kewisi KWS-20 USB power meter. As a baseline for energy consumption, the Raspberry Pi Zero W draws approximately 0.36 W while in idle mode.

For consistency, the implementation follows the NIST API, where key pair

excluded due to incompatibility with ARMv6 architecture. Nonetheless, SDitH, MiRitH and MIRA (before their merge into MIRA) are drawn from Round 1 (https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures).

[2]https://csidh.isogeny.org/software.html .

[3]https://github.com/zinho02/lorawan-pq .

generation, signing, and verification are tested using the methods *crypto_sign_keypair*, *crypto_sign*, and *crypto_sign_open*, respectively. All tests are performed using the reference implementation of each algorithm, without optimizations.

Table 1 presents the performance metrics for CSIDH. The similarity between public key generation and shared secret computation stems from both operations relying on the application of a CSIDH group action [Castryck et al. 2018]. The performance evaluation of the post-quantum signature algorithms, sorted by energy consumption, and the ToA of the messages exchanged in the procedures are detailed in Figure 3. These results are analyzed in the following subsections.

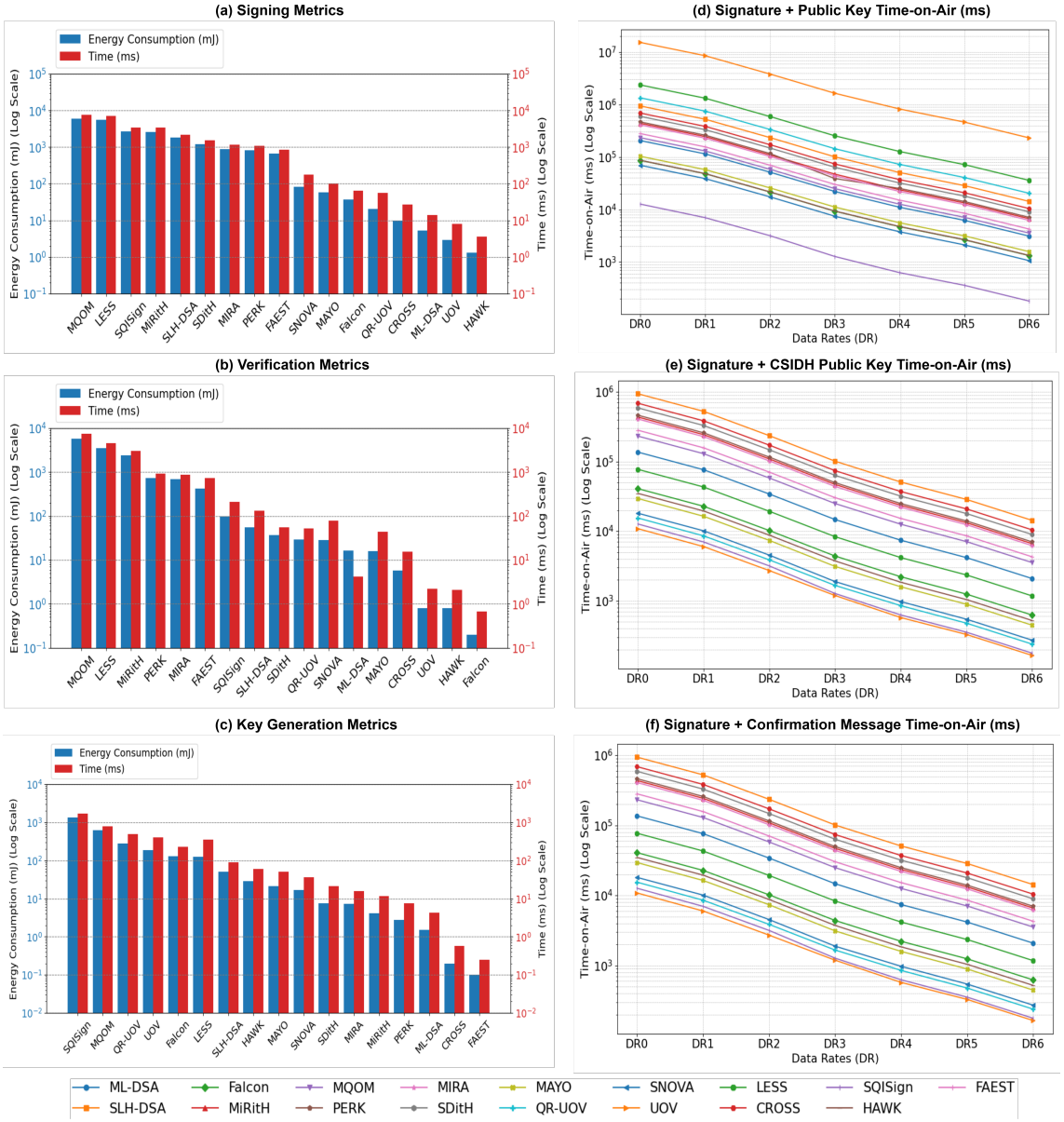

Figure 3. Post-Quantum Signature Algorithms Metrics and Time-on-Air of messages

5.1.1. Root Key Renewal Procedure

As previously discussed, the RKR procedure incorporates CSIDH operations along with the signing and verification steps of post-quantum digital signature algorithms. When comparing the performance of CSIDH with that of the signature schemes, it becomes evident that CSIDH still trails in execution time and energy efficiency. However, this tradeoff is offset by its remarkably compact public key, which is only 64 bytes.

Figures 3a and 3b show the signing and verification performance of the signature algorithms. As expected, the energy and time graphs follow a very similar pattern, with a minor exception observed in ML-DSA verification time. MQOM and LESS stand out as the least efficient in terms of both execution time and energy consumption, rendering them less suitable for constrained environments. Moreover, LESS suffers from both a large public key and signature size, while MQOM, although benefiting from a small public key, produces a relatively large signature.

Some algorithms show excellent performance but are limited by key or signature size. UOV, despite its excellent computational efficiency, is hindered by an extremely large public key, only partially offset by its relatively small signature. Similarly, CROSS features a compact public key but is burdened by a large signature. In contrast, HAWK and ML-DSA emerge as top performers, offering low execution time and energy consumption with a particularly strong balance of efficiency and compact public key and signature sizes, as shown in Table 2.

5.1.2. Key Pair Update Procedure

In the KPU procedure, key generation and verification are the dominant operations for both the ED and Join Server (JS). While Figure 3b was previously discussed, Figure 3c reveals a similar pattern in execution time and energy consumption. Notably, SQISign and MQOM stand out as the most resource-intensive for key generation, requiring significantly more time and energy than other candidates. MQOM is further constrained by its large public key. Despite its computational demands, SQISign remains a viable option due to its minimal public key and signature sizes.

Conversely, FAEST and CROSS are among the most efficient, demonstrating low execution time and energy use. Although CROSS produces a relatively large signature, both algorithms benefit from compact public keys, as detailed in Table 2.

5.2. Time-on-Air

To estimate the total ToA of each message involved in the RKR and KPU procedures, the transmitted data is segmented into packets that comply with the maximum payload limitations of the LoRaWAN specification [LoRa Alliance Technical Committee 2017]. The ToA of each procedure is then computed as the sum of the ToA required for each individual packet. For simplicity, it is assumed that upstream and downstream transmissions have equivalent ToA — as may be the case in symmetric link scenarios. Nonetheless, this approach effectively provides a lower bound estimate for the total ToA of each message.

The ToA values are obtained using The Things Network (TTN) LoRaWAN Airtime Calculator [The Things Network 2025], which adheres to the standard pa-

rameters defined for LoRaWAN packets under the EU863-870 frequency plan [LoRa Alliance Technical Committee Regional Parameters Workgroup 2022].

5.2.1. Root Key Renewal Procedure

Renewing the root keys requires accounting for the ToA associated with transmitting both the signature with the public key and the CSIDH public key with the signature. The ToA of the KeyID is excluded due to its negligible impact. Figures 3d and 3e illustrate the ToA of these two main components. From DR6 downward, the ToA roughly doubles with each step, resulting in a notable disparity between DR0 and DR6.

As shown in Figure 3d, transmitting the signature together with the public key leads to impractically high ToA values at low Data Rates (DRs), often nearing 70 seconds. UOV, QR-UOV, and LESS remain inefficient even at DR6, primarily due to their large public key sizes. In contrast, SQISign, SNOVA, Falcon, HAWK, and MAYO show practical ToA values, typically around 1 second at higher DRs, thanks to smaller public key and signature sizes. SQISign performs particularly well, reaching as low as 200 ms and maintaining efficiency even at lower DRs.

When sending the signature with the CSIDH public key, as shown in Figure 3e, CROSS, SDitH and SLH-DSA perform poorly as a result of their large signature sizes. Meanwhile, schemes with smaller signatures, such as UOV and QR-UOV, achieve significant improvements. At DR6, UOV, SQISign, QR-UOV, SNOVA, MAYO, HAWK, and Falcon all deliver ToA values well under one second.

5.2.2. Key Pair Update Procedure

To update its key pair, the ED must generate a new one, transmit the new public key — signed with the old private key — to the PKI server, and, upon receiving the signed confirmation, safely replace the old key pair. The associated ToA is shown in Figures 3d and 3f. Building on the analysis from Figure 3d, Figure 3f further highlights the impracticality of SLH-DSA, CROSS, and SDitH, which show ToA values of at least 8 seconds even in the best cases, reinforcing that large signatures are impractical in this context. In contrast, UOV, SQISign, QR-UOV, SNOVA, MAYO, HAWK, and Falcon confirm their practicality, all staying below 600 ms at DR6, consistent with the trends observed in Figure 3e.

5.3. Total Procedure Time and Energy Consumption

To estimate the total time and energy consumption of each procedure from the ED's perspective, both the ToA of transmitted messages and the execution time and energy of local operations (e.g., signature verification) are taken into account. The total procedure energy consumption is obtained by summing the energy of all relevant local computations, as defined in Equations (2) and (4). Likewise, the total procedure time is calculated by adding the ToA of each message to the duration of corresponding ED local operations, as detailed in Equations (1) and (3). Definitions of all associated terms are provided in Table 3.

Table 3. Symbol Definitions

Symbol	Definition	Symbol	Definition
T_{csidh_pk}	CSIDH public key generation time	E_{csidh_pk}	CSIDH public key generation energy consumption
T_{csidh_share}	CSIDH shared secret computation time	E_{csidh_share}	CSIDH shared secret generation energy consumption
$T_{pq\_key\_gen}$	Post-quantum key pair generation time	$E_{pq\_key\_gen}$	Post-quantum key pair generation energy consumption
T_{pq_sig}	Post-quantum signing time	E_{pq_sig}	Post-quantum signing energy consumption
T_{pq_ver}	Post-quantum verification time	E_{pq_ver}	Post-quantum verification energy consumption
$T_{sig\_csidh\_toa}$	ToA for post-quantum signature and CSIDH public key	$E_{total\_root\_keys}$	Total RKR procedure energy consumption
$T_{pk\_sig\_toa}$	ToA for post-quantum public key and signature	$E_{total\_key\_pair}$	Total KPU procedure energy consumption
$T_{pki\_sig\_ack\_toa}$	ToA for post-quantum signature and confirmation message		
$T_{total\_root\_keys}$	Total RKR procedure time		
$T_{total\_key\_pair}$	Total KPU procedure time		

Factors with negligible impact — such as TCP/IP transmission delays, Ascon-Hash256 evaluations, CSIDH secret key generation, KeyID ToA, and computations performed on the remote end — are excluded from the analysis to focus on the dominant contributors to time and energy costs. To yield a conservative lower-bound estimate, the ED is assumed to operate in half-duplex mode, sharing the same frequency for uplink and downlink, thereby enforcing serialized message transmission and reception.

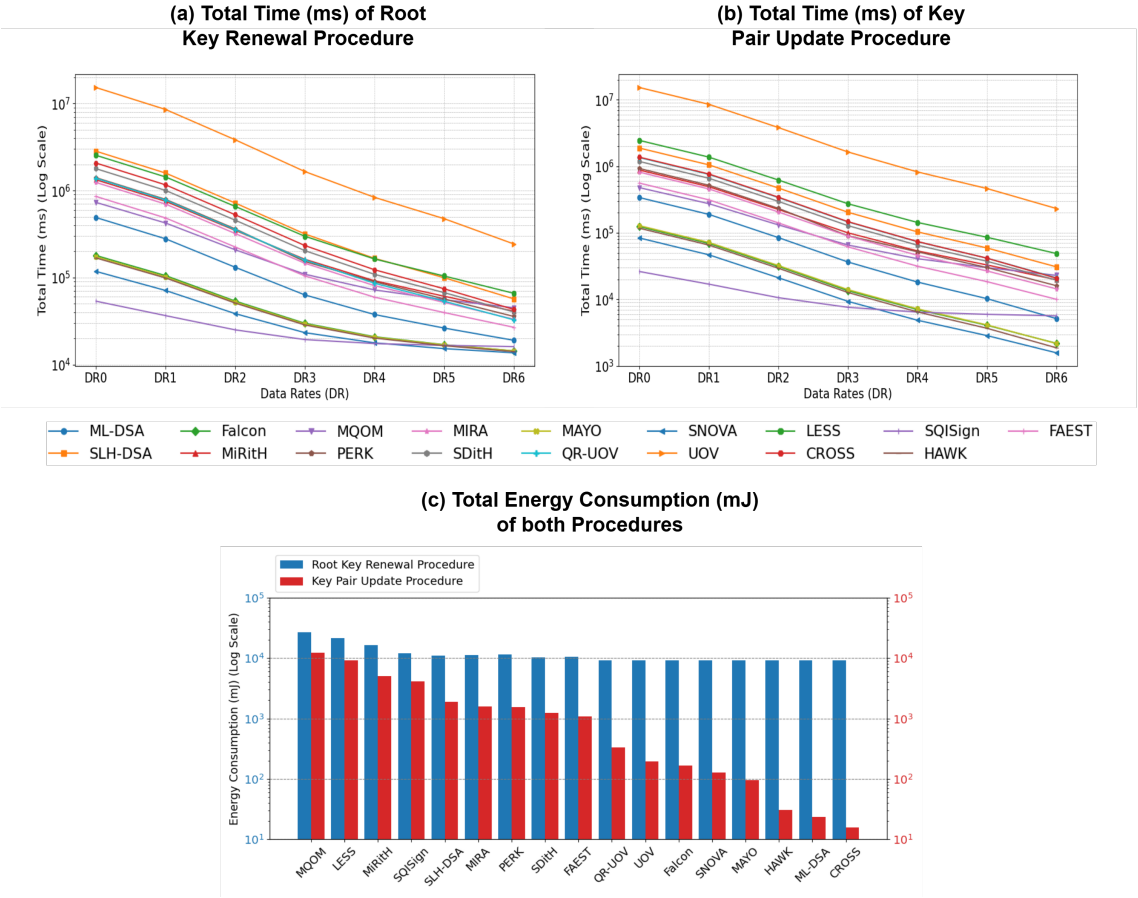

Figure 4. Total Time (ms) and Energy Consumption (mJ) of both Procedures

5.3.1. Root Key Renewal Procedure

The total execution time and energy consumption of the RKR procedure are given by Equations (1) and (2), respectively:

$$T_{total\_root\_keys} = T_{pk\_sig\_toa} + T_{csidh\_pk} + T_{pq\_sig} + 2 \cdot T_{sig\_csidh\_toa} + 2 \cdot T_{pq\_ver} + T_{csidh\_share} \quad (1)$$

$$E_{total\_root\_keys} = E_{csidh\_pk} + E_{pq\_sig} + 2 \cdot E_{pq\_ver} + E_{csidh\_share} \quad (2)$$

Figure 4a shows that LESS, UOV, and SLH-DSA remain unusable, with total execution times over 57 seconds — even at the best DRs. Meanwhile, MAYO, Falcon, HAWK, SQISign, and SNOVA show strong results, with ToAs under 17 seconds even at DR5. This performance is mainly due to a balanced mix of efficient local operations and compact public key and signature sizes, which together minimize overall ToA.

In the context of the RKR procedure shown in Figure 4c, the results indicate that most algorithms exhibit similar energy consumption patterns. Notably, MQOM, LESS, and MiRitH show the highest energy consumption, reaching up to 27 J in the worst case. In contrast, the remaining algorithms fall within a narrower range, approximately 10 J, where most of the energy consumption is attributed to CSIDH computations.

5.3.2. Key Pair Update Procedure

The total execution time and energy consumption for the KPU procedure are given by Equations (3) and (4), respectively:

$$T_{total\_key\_pair} = T_{pq\_key\_gen} + T_{pq\_sig} + T_{pk\_sig\_toa} + T_{pki\_sig\_ack\_toa} + T_{pq\_ver} \quad (3)$$

$$E_{total\_key\_pair} = E_{pq\_key\_gen} + E_{pq\_sig} + E_{pq\_ver} \quad (4)$$

Figure 4b shows that the KPU procedure introduces significantly less ToA overhead compared to RKR. Nonetheless, performance trends remain largely consistent across both procedures, with most algorithms maintaining their relative efficiency or inefficiency. This is due to the shared components still dominating total execution time, despite the lower overhead from key pair generation. One notable exception is SQISign, which sees a performance drop from DR4, as its intensive local operations begin to outweigh reduced transmission costs.

Similarly, Figure 4c confirms that the KPU procedure results in substantially lower energy consumption than RKR. However, the gap between efficient and inefficient algorithms becomes more apparent. MQOM, LESS, and MiRitH are the least energy-efficient, consuming up to 12 J. Conversely, several algorithms use under 1 J, with CROSS, ML-DSA, and HAWK standing out for achieving as little as 10 mJ in ideal conditions.

The consistent performance observed in the RKR procedure highlights the practicality of MAYO, Falcon, HAWK, SNOVA, and SQISign for LoRaWAN deployments. Although SQISign exhibits higher energy use during KPU, it remains suitable for low DR scenarios. Furthermore, given that RKR and KPU are infrequent operations, their overhead is acceptable considering the significant security enhancements they offer.

6. Conclusion and Future Works

The IoT is transforming industries through smart automation, with LoRaWAN emerging as a leading LPWAN standard for long-range, low-power communication in constrained settings. However, its static root keys — embedded at manufacture and never updated — make it vulnerable to key compromise. Additionally, the rise of quantum computers threatens traditional cryptography, highlighting the need for post-quantum security.

This paper addresses both issues by proposing a quantum-safe solution to the static root key vulnerability. The approach integrates a post-quantum authenticated key exchange using CSIDH with NIST PQC digital signatures. This enables dynamic root key updates via a Root Key Renewal (RKR) procedure. Once new root keys are set, the standard Join/Rejoin Procedure derives fresh session keys, maintaining security for ongoing communications. To avoid shifting the static nature to the new signature key pairs, a complementary Key Pair Update (KPU) procedure is introduced to ensure their periodic renewal.

To evaluate feasibility in constrained settings, tests were conducted to measure execution time, energy use, Time-on-Air (ToA), and total procedure duration and energy consumption. The results show that while some post-quantum algorithms offer fast computations, their large keys and signatures greatly increase transmission time — as seen with UOV and CROSS. In contrast, HAWK, SNOVA, MAYO, and Falcon strike a strong balance across all metrics, emerging as practical candidates. Notably, SQISign, despite higher computational demands, provides compact key and signature sizes that fit within a single LoRaWAN packet — ideal for bandwidth-limited and low-data-rate scenarios where others fall short. The findings further highlight that better overall performance stems from smaller key and signature sizes. Ultimately, the overhead remains acceptable given the infrequent nature of both procedures and the significant security benefits they provide.

Future work should formally verify both procedures and implement them in real-world deployments across various classes of End Devices (EDs). Optimization strategies for execution time and energy consumption — especially for CSIDH, such as fast finite field arithmetic — should be explored and evaluated against this baseline. Additionally, it will be important to analyze potential threats such as replay and DDoS attacks.

References

Alagic, G. (2025). Status report on the fourth round of the nist post-quantum cryptography standardization process. Technical report, National Institute of Standards and Technology.

Alagic, G., Bros, M., Ciadoux, P., Cooper, D., Dang, Q., Dang, T., Kelsey, J., Lichtinger, J., Liu, Y.-K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Silberg, H., Smith-Tone, D., and Waller, N. (2024). Status report on the first round of the additional digital signature schemes for the nist post-quantum cryptography standardization process. Technical report, National Institute of Standards and Technology (U.S.).

Almuhaya, M. A. M., Jabbar, W. A., Sulaiman, N., and Abdulmalek, S. (2022). A survey on lorawan technology: Recent trends, opportunities, simulation tools and future directions. *Electronics*, 11(1):164.

Butun, I., Pereira, N., and Gidlund, M. (2018). Security risk analysis of lorawan and future directions. *Future Internet*, 11(1):3.

Castryck, W., Lange, T., Martindale, C., Panny, L., and Renes, J. (2018). CSIDH: An efficient post-quantum commutative group action. Cryptology ePrint Archive, Paper 2018/383.

Chen, X., Lech, M., and Wang, L. (2021). A complete key management scheme for lorawan v1.1. *Sensors*, 21(9):2962.

Dobraunig, C., Eichlseder, M., Mendel, F., and Schläffer, M. (2021). Ascon v1.2: Lightweight authenticated encryption and hashing. *Journal of Cryptology*, 34(3).

Donmez, T. C. M. and Nigussie, E. (2019). Key management through delegation for lorawan based healthcare monitoring systems. In *2019 13th International Symposium on Medical Information and Communication Technology (ISMICT)*, page 1–6. IEEE.

Figlarz, G. R. and Hessel, F. P. (2024). Enhancement in lorawan's security with post-quantum key encapsulation method. In *2024 IEEE 10th World Forum on Internet of Things (WF-IoT)*, page 804–809. IEEE.

Hayati, N., Ramli, K., Windarta, S., and Suryanegara, M. (2022). A novel secure root key updating scheme for lorawans based on ctr_aes drbg 128. *IEEE Access*, 10:18807–18819.

Issac, K., Pranay, G., Bharanidharan, N., and Rajaguru, H. (2020). A study on real world implementation and future trends of internet of things. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 357–361.

Jao, D. and De Feo, L. (2011). *Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies*, page 19–34. Springer Berlin Heidelberg.

LoRa Alliance Technical Committee (2017). Lorawan 1.1 specification. Technical report, LoRa Alliance. Available at: `https://lora-alliance.org/resource_hub/lorawan-specification-v1-1`. Accessed: March 2025.

LoRa Alliance Technical Committee Regional Parameters Workgroup (2022). Lorawan regional parameters rp002-1.0.4. Available at: `https://`

resources.lora-alliance.org/technical-specifications/rp002-1-0-4-regional-parameters. Accessed: March 2025.

Marlind, F. and Butun, I. (2020). Activation of lorawan end devices by using public key cryptography. In *2020 4th Cyber Security in Networking Conference (CSNet)*, page 1–8. IEEE.

Mekki, K., Bajic, E., Chaxel, F., and Meyer, F. (2018). Overview of cellular lpwan technologies for iot deployment: Sigfox, lorawan, and nb-iot. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, page 197–202. IEEE.

Milani, S. and Chatzigiannakis, I. (2021). Design, analysis, and experimental evaluation of a new secure rejoin mechanism for lorawan using elliptic-curve cryptography. *Journal of Sensor and Actuator Networks*, 10(2):36.

Ntshabele, K., Isong, B., Gasela, N., and Abu-Mahfouz, A. M. (2022). A comprehensive analysis of lorawan key security models and possible attack solutions. *Mathematics*, 10(19):3421.

Papatsaroucha, D., Astyrakakis, N., Pallis, E., Grammatikis, P. I. R., Sarigiannidis, P. G., and Markakis, E. K. (2024). A cloud-based key rolling technique for alleviating join procedure replay attacks in lorawan-based wireless sensor networks. In *2024 IEEE International Conference on Big Data (BigData)*, page 2811–2820. IEEE.

Qadir, J., Butun, I., Gastaldo, P., Aiello, O., and Caviglia, D. D. (2023). Mitigating cyber attacks in lorawan via lightweight secure key management scheme. *IEEE Access*, 11:68301–68315.

Ribeiro, V., Filho, R. H., and Ramos, A. (2019). A secure and fault-tolerant architecture for lorawan based on blockchain. In *2019 3rd Cyber Security in Networking Conference (CSNet)*, page 35–41. IEEE.

Robert, D. (2023). *Breaking SIDH in Polynomial Time*, page 472–503. Springer Nature Switzerland.

Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509.

Sonmez Turan, M., McKay, K., Chang, D., Bassham, L. E., Kang, J., Waller, N. D., Kelsey, J. M., and Hong, D. (2023). Status report on the final round of the nist lightweight cryptography standardization process. Technical report, National Institute of Standards and Technology (U.S.).

The Things Network (2025). Lorawan airtime calculator. Available at: https://www.thethingsnetwork.org/airtime-calculator. Accessed: March 2025.

Tsai, K.-L., Chen, L.-W., Leu, F.-Y., and Wu, C.-T. (2022). Two-stage high-efficiency encryption key update scheme for lorawan based iot environment. *Computers, Materials & Continua*, 73(1):547–562.

You, I., Kwon, S., Choudhary, G., Sharma, V., and Seo, J. T. (2018). An enhanced lorawan security protocol for privacy preservation in iot with a case study on a smart factory-enabled parking system. *Sensors*, 18(6):1888.