Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2025.DOI

LiteQSign: Lightweight and Quantum-Safe Signatures for Heterogeneous IoT Applications

ATTILA A. YAVUZ[1], SALEH DARZI[1], and SAIF E. NOUMA[1]
[1]University of South Florida, 3720 Spectrum Blvd, Interdisciplinary Research Building (IDR)-400, Tampa, Florida, USA 33612, (e-mail: {attilaayavuz, salehdarzi, saifeddinenouma}@usf.edu)

This work is supported by National Science Foundation NSF- SNSF 2444615 and Cisco Research Award (220159).

ABSTRACT The rapid proliferation of resource-constrained IoT devices across sectors like healthcare, industrial automation, and finance introduces major security challenges. Traditional digital signatures, though foundational for authentication, are often infeasible for low-end devices with limited computational, memory, and energy resources. Also, the rise of quantum computing necessitates post-quantum (PQ) secure alternatives. However, NIST-standardized PQ signatures impose substantial overhead, limiting their practicality in energy-sensitive applications such as wearables, where signer-side efficiency is critical.

To address these challenges, we present LightQSign (`LiteQS`), a novel lightweight PQ signature that achieves near-optimal signature generation efficiency with only a small, constant number of hash operations per signing. Its core innovation enables verifiers to obtain one-time hash-based public keys without interacting with signers or third parties through secure computation. We formally prove the security of `LiteQS` in the random oracle model and evaluate its performance on commodity hardware and a resource-constrained 8-bit AtMega128A1 microcontroller. Experimental results show that `LiteQS` outperforms NIST PQ standards with lower computational overhead, minimal memory usage, and compact signatures. On an 8-bit microcontroller, it achieves up to 1.5–$24\times$ higher energy efficiency and 1.7–$22\times$ shorter signatures than PQ counterparts, and 56–$76\times$ better energy efficiency than conventional standards–enabling longer device lifespans and scalable, quantum-resilient authentication.

INDEX TERMS Digital Signatures, Internet of Things, Lightweight Authentication, Post-Quantum Security

I. INTRODUCTION

The proliferation of resource-constrained IoT devices, coupled with their extensive integration into critical domains such as healthcare, industrial automation, and financial services, introduces significant security challenges. While these devices deliver substantial utility and advancements, their operation in challenging environments and inherent limitations make them the most vulnerable link in the security chain. With billions of IoT devices interconnected across networks, many function in environments that process sensitive or personal data, including healthcare records [1], military communications [2], and security logs [3], [4], while executing essential operations such as unlocking smart doors, regulating industrial machinery [5], and administering medical treatments [6]. For instance, compromised authentication undermines the integrity of data from implantable devices, rendering them ineffective and potentially endangering patients, such as failing to correct a slow heartbeat in time [7]. Therefore, ensuring device authenticity and data integrity is essential for protecting networks against malicious entities, preserving reliable data transmission, and preventing unauthorized access to critical control systems. Moreover, IoT devices rely on periodic updates to mitigate vulnerabilities and enhance functionality; verifying the authenticity and integrity of these updates is vital to prevent unauthorized or malicious software installation and to maintain overall system security [3], [8], [9].

A "*digital signature*" serves as the cornerstone cryptographic solution for ensuring authentication and data integrity while providing essential features such as non-repudiation, public verifiability, and scalability [10], [11]. These properties enable IoT systems to operate securely and efficiently, mitigating risks of unauthorized access, data breaches, device manipulation, and network disruptions. Given the diverse operational constraints of IoT applications–particularly in computation and energy efficiency–authentication mechanisms must be tailored accordingly. Here, we focus on *IoT-enabled applications that prioritize lightweight signing due to resource constraints, while slower verification is acceptable*

on backend systems. Numerous real-world IoT use cases align with this paradigm. Examples include environmental and industrial sensors that periodically transmit data, such as temperature or air quality, to central servers, while smart utility meters (e.g., energy, water, gas) frequently report usage data [12]–[14]. Wireless sensor networks with verifier-side delay tolerance [15] and wearable health devices continuously collect biometric metrics, forwarding them to paired devices or the cloud [16]–[18]. Remote IoT nodes, including wildlife trackers and agricultural sensors [19], operate in power-constrained environments, transmitting critical updates. RFID/NFC-based asset tracking systems enable identification with resourceful verifiers [20], [21], while smart city sensors for traffic, pollution, and waste management autonomously collect and transmit data on low power, prioritizing lightweight signing and deferred verification [22].

A. MAIN DESIGN CONSIDERATIONS

Designing practical authentication solutions for heterogeneous IoT enabled applications, where the signer is resource constrained and the verifier is resourceful, requires addressing the following key considerations:

(I) Post-Quantum Security: The rise of quantum computing threatens traditional digital signatures based on number-theoretic assumptions (e.g., factorization, discrete logarithm) [23]–[25], necessitating long-term security solutions with PQ guarantees for IoT applications handling sensitive data and critical operations. Many IoT devices have long lifespans and cannot easily transition to new cryptographic frameworks once deployed, making immediate PQ adoption essential [26]. For instance, updating security measures on heart implants requires surgery, and medical audit logs must remain verifiable for decades [16], while upgrading mass-produced smart meters at scale is costly and impractical [27]. Moreover, resource constraints such as limited processing power, memory, and battery life (e.g., 8-bit microcontrollers) further exacerbate PQ deployment. Addressing these challenges is critical for securing IoT-enabled environments, ensuring long-term resilience against emerging quantum threats [28].

(II) Computationally Efficient Signature Generation with Minimal Energy Usage: The primary design consideration is lightweight signature generation with signer-efficient operations and minimal energy consumption. This is especially critical for battery-powered embedded IoT devices [29], where energy efficiency directly affects system longevity. For instance, in wearables, prolonged battery life enhances usability [4], while for implantable devices, it directly impacts patient well-being, as replacements may require surgical intervention [29]. Hence, the underlying cryptographic mechanism must minimize energy consumption to extend device longevity. However, even standard digital signatures based on conventional security, such as Elliptic Curve Cryptography (ECC), have been shown to degrade battery life in low-end IoT devices [30], [31]. This challenge is further intensified when PQ security is considered [26], [32].

(III) Minimal Cryptographic Memory and Bandwidth Usage: *(i)* Embedded IoT platforms, such as the 8-bit ATxmega128A1 MCU with only 128 *KB* of flash memory[1], demand strict memory efficiency. In devices with severe memory and bandwidth constraints (e.g., heart implants, wearables), NIST-standardized signatures like ML-DSA and SLH-DSA–requiring approximately $6.2KB$ and $71.9KB$ for combined signature, private, and public keys–are unsuitable for deployment. Thus, lightweight signatures must feature compact keys and minimal memory expansion during computation. (ii) Code size is another constraint, where schemes relying on simple primitives (e.g., hashing) over computationally intensive operations (e.g., EC scalar multiplication [33], sampling [34]) offer reduced energy usage and implementation overhead. (iii) Larger signature sizes increase memory and bandwidth demands and significantly drain the battery during transmission [35]. These limitations make large post-quantum secure signatures particularly challenging for resource-constrained IoT devices.

(IV) Security Assumptions and Robustness (i) Minimal Security Assumptions: While some schemes improve efficiency by assuming semi-honest, non-colluding servers or secure enclaves [3], [28], the security-sensitive nature of IoT-enabled applications necessitates avoiding such dependencies. Eliminating extra architectural assumptions strengthens long-term trust, not only through PQ cryptographic guarantees but also by enhancing resilience against emerging threats. *(ii)* Robustness Against Side-Channel Attacks: PQ signatures involve Gaussian sampling [36], rejection sampling [34], and complex arithmetic [32], [37], making them vulnerable to side-channel attacks [38], [39]. These risks are amplified in embedded architectures due to the difficulty of implementing countermeasures [4]. Additionally, low-end IoT devices often generate low-quality random numbers, exposing them to cryptographic vulnerabilities [40]. Addressing these challenges is essential for securing IoT systems. *(iii)* Scalability and Interoperability: Ensuring seamless communication among diverse devices in heterogeneous IoT ecosystems requires an authentication scheme that efficiently scales across millions of interconnected, resource-limited devices. Given the challenges of key management in large-scale IoT networks with constrained connectivity, maintaining constant-size public keys and lightweight mechanisms for key distribution, renewal, and revocation is essential.

B. RELATED WORK AND LIMITATIONS OF THE STATE-OF-THE-ART

This section examines state-of-the-art signatures that address key considerations for IoT-enabled applications, with a focus on schemes offering PQ security, computationally efficient signing, and compact public key and signature sizes. Given the extensive range of proposed signatures, we first provide a brief overview of prominent conventional signatures before shifting to PQ alternatives, including both standardized schemes and those optimized for signing efficiency. Our se-

[1]https://www.microchip.com/en-us/product/atxmega128a1

TABLE 1: Comparison of Digital Signature Schemes for IoT: Vulnerabilities, Overheads, and Limitations

	Scheme	Hardness Assumption	Vulnerability			Overhead			Limitation	
			QA	SCA	Other				EoI	EmI
Classical	ECDSA [41]	Elliptic curve	✓	✓	-				✗	✓
	Ed25519 [42]		✓	✓	-				✗	✓
	RSA [43]	Factorization	✓	-	-				✗	✗
	BLS [44]	Pairing-based	✓	-	-				✗	✗
Post-Quantum	ML-DSA [34]	Lattice-based	-	✓	-				✗	✗
	FN-DSA [36]		-	✓	-				✗	✗
	BLISS [37]		-	✓	-				✗	✓
	ANT [3]		-	-	NCD-S				✓	✓
	SLH-DSA [45]	Hash-based	-	-	-				✗	✗
	XMSSMT [46]		-	-	-				✗	✗
	HASES [11]		-	-	TEE-S				✓	✓
	MAYO [47]	Multivariate	-	-	PA				✗	✗

QA, SCA, PA, EoI, and EmI denote quantum attacks, side-channel attacks, polynomial attacks, ease of implementation, and embedded implementation on low-end 8-bit devices, respectively. , , and indicate transmission, memory, and computation overhead, respectively. , , and indicate low, medium, and high energy usage on IoT devices. ANT [3] relies on non-colluding distributed servers to supply verifiers with one-time public keys. HASES [11] relies on TEE-enabled single, thus suffer from single-root of trust.

lected digital signature schemes are summarized in Table 1 and further elaborated as follows:

Conventional lightweight signatures: These schemes offer efficient signature generation, compact key sizes, and additional security guarantees [31], [48], [49]. Among conventional signatures, EC Schnorr-based schemes [33] are particularly efficient compared to pairing-based [44] and factorization-based [43] alternatives. For instance, a recent EC-based scheme [31] enhances signer efficiency and supports single-signer signature aggregation, while Chen et al. [49] integrate confidentiality with authentication. However, despite their advantages, none of these ECC-based schemes or similar conventional signatures provide PQ security.

Standard PQ signatures: NIST has recently standardized two PQ digital signature schemes: the Module-Lattice-Based Digital Signature Standard (ML-DSA, FIPS 204) [34] and the Stateless Hash-Based Digital Signature Standard (SLH-DSA, FIPS 205) [45], representing lattice-based and hash-based approaches, respectively [50]. The following discusses these domains in detail.

Known for their minimal intractability assumptions, hash-based signatures provide strong security guarantees. The stateless SLH-DSA [45], derived from a variant of the one-time signature scheme HORS [51], employs a hyper-tree structure for multiple-time signatures. While SLH-DSA ensures PQ security with strong assumptions, its signing time and signature size are orders of magnitude slower and larger than ECDSA, making it unsuitable for resource-constrained IoT devices. Similarly, stateful hash-based signatures (e.g., RFC-standardized XMSS-MT [52] and LMS [53]), built on HORS variants like W-OTS [54], offer comparable security and forward security. However, their state management requirements, high computational cost, and memory demands render them impractical for low-end IoT devices.

Based on module lattice problems (e.g., LWE [34]), lattice-based signatures offer a balanced trade-off between signing and verification efficiency. NIST-selected schemes, ML-DSA [34] and FN-DSA [36], achieve smaller signatures and faster signing than SLH-DSA. However, they remain unsuitable for resource-constrained IoT devices due to their computational complexity and larger signature sizes compared to conventional signatures. Additionally, techniques such as Gaussian and rejection sampling introduce vulnerabilities to side-channel attacks [38]. To date, no open-source lattice-based signature implementation is optimized for highly constrained devices like 8-bit microcontrollers, except for BLISS [37], which was not selected as a NIST PQC standard [55] and is susceptible to side-channel attacks [39].

Additional PQ Signatures for Standardization: To diversify PQ signature standards, NIST launched an additional signature competition alongside its standardized schemes, emphasizing fast verification and compact signatures. Currently in its second round, it includes submissions across various PQC categories, such as code-based [56], multivariate-based [47], and symmetric-based [57] signatures. For instance, Shim et al. [32] improves upon NIST PQC standards with efficient signing and smaller signatures but suffers from large private key and code sizes, making it impractical for IoT devices. Its $12.6KB$ private key is an order of magnitude larger than ECDSA, and its implementation occupies 62.6% of the total flash memory on an ATxmega128A1 MCU, imposing significant memory overhead. Despite PQC advancements, computational and memory constraints remain major obstacles for low-end IoT devices, particularly those operating on 8-bit MCUs [55]. Also, many multivariate signatures not only demand extensive memory and stack resources but are also susceptible to polynomial-time attacks [58].

Lightweight PQ Signatures with Additional Assumptions: These schemes achieve highly efficient signature generation but rely on additional assumptions [3], [28], [60]. For example, ANT [3], a lattice-based signature, delegates costly commitment generation to distributed, non-colluding, semi-honest servers [61]. However, verifiers must interact with these servers before verification, introducing potential network delays and outage risks. Another approach leverages Trusted Execution Environment (TEE)-assisted signatures, offloading computational overhead to TEE-enabled servers [11], [28], [60]. For instance, HASES [11] and its extension

TABLE 2: Performance comparison of LiteQS and its counterparts on commodity hardware

Scheme	Signing Time (μs)	Private Key (KB)	Signature Size (KB)	Ver Time (μs)		Verifier Storage (KB)		Total Storage (GB)	SCB
				Online	Offline	Public Key	Certificate		
BLISS-I [37]	244.97	2.00	5.6	25.21		7.00	5.6	12.6	×
ML-DSA-II [59]	93.76	2.29	2.36	18.73		1.28	2.36	3.75	×
FN-DSA-512 [36]	184.74	1.29	0.65	32.16		0.88	0.65	1.53	×
SLH-DSA [45]	5,441.58	0.13	32.63	549.63		0.06	32.63	32.69	×
XMSS^{MT} [46]	10,682.35	3.11	2.61	2,098.84		0.75	2.61	3.36	×
LiteQSign	**4.81**	**0.02**	**0.25**	**1.91s**		**9.42 MB**		**9.42 MB**	✓

The private/public key, signature, and certificate sizes are in KB. SCB denotes a simple code base. LiteQS and NIST PQC candidates use architecture-specific optimizations (i.e., AESNI, AVX2 instructions). For XMSS^{MT}, we choose the XMSST_MT_SHA2_20_256 variant. For SLH-DSA, we set $n = 256, h = 63, d = 9, b = 12, k = 29, w = 16$. The total verifier storage denotes the storage required to verify ($J = 2^{30}$) signatures for ($N = 2^{20}$) signers. LiteQS incurs an offline PKContr cost of 41.22 seconds.

[28], derived from the one-time HORS [51], use a single TEE-enabled cloud server to issue one-time public keys. However, reliance on a centralized TEE server introduces key escrow risks and a single point of trust. Due to these assumptions, such signatures may not be ideal for IoT-enabled applications that require adherence to traditional public key settings.

Given the limitations of existing signatures and the gap in achieving all desirable properties for IoT applications, there is a critical need for efficient PQ signatures that balance performance and security while enabling signer-optimal generation and deferred verification. This work explores the following key research questions: *(i) Can an efficient PQ signature scheme be designed with optimal signature generation while meeting IoT constraints on memory, processing, and bandwidth? (ii) Is it possible to achieve energy-efficient signing without introducing unconventional or risky assumptions for verifiers? (iii) Can these requirements be met in a scalable multi-user setting for IoT networks?*

C. OUR CONTRIBUTION AND DESIRABLE PROPERTIES

LiteQS extends the one-time HORS scheme into a multi-time signature while preserving its high signing efficiency. It offers optimal signer-side performance with minimal computation, resulting in excellent energy efficiency and low overhead. The scheme maintains a constant-size, compact private key derived from a short seed, produces small signatures, and supports online-offline verification using a single master key. Moreover, it operates without requiring additional security assumptions such as non-colluding parties or distributed trusted servers. The desirable properties of LiteQS are depicted in TABLE 2 and outlined as follows:

• ***Signer-Optimal Computation with High Energy Efficiency***: LiteQS achieves *optimal efficiency* with only a small, constant number (e.g., 16) of Pseudo-Random Function (PRF) calls per signing, outperforming ML-DSA-II and FN-DSA-512 by $19.5\times$ and $1130\times$, respectively, and exceeding the speed of top ECC-based signatures. This translates into substantial energy savings (see Table 4 and Section VI).

• ***Minimal Memory Requirements and Bandwidth Efficiency***: LiteQS uses a single 128-bit seed as the private key and transmits only one 256-byte HORS signature per message, making it the most compact among PQ alternatives (see Table 4). It avoids heavy operations like EC scalar multiplication and lattice sampling, relying instead on a few PRF calls and one hash, while ensuring standard compliance and facilitating an efficient transition to PQC [28].

• ***Advanced Security Features and Robustness***: *(i)* LiteQS adheres to the standard public key model, avoiding unconventional assumptions such as non-colluding or trusted key distribution servers [3], which introduce architectural risks. *(ii)* Unlike lattice-based schemes prone to side-channel and timing attacks due to Gaussian and rejection sampling [38], LiteQS relies solely on symmetric cryptographic primitives, eliminating these vulnerabilities. Also, its deterministic signature generation prevents weaknesses from poor random number generators, a common issue in resource-limited IoT devices.

• ***Compact Multi-User Storage and Online/Offline Verification***: LiteQS offers a scalable solution by allowing verifiers to derive one-time public keys from a constant-size master public key, removing the need for storing per-user keys or certificates—even for large-scale deployments (e.g., 2^{20} users). Public keys can be precomputed or generated on-demand before verification. Though key construction involves encrypted function evaluations, it can be performed independently by the verifier or offloaded to a resourceful cloud server, significantly minimizing its practical impact.

• ***Potential Use Cases, Strengths, and Limitations***: LiteQS is well-suited for lightweight authentication in resource-constrained IoT applications, particularly those with verification-delay tolerance. One compelling use case is secure logging and system auditing, where authentication of log streams enables early detection of malware and intrusions [62]–[64]. In such scenarios, memory-limited devices can offload logs to the cloud for long-term storage and periodic integrity verification [63]. LiteQS minimizes the signer's computational and energy burden by requiring only one hash-based signature per log entry, while deferring the heavier verification process to resourceful back-end systems. The primary strength of LiteQS lies in its near-optimal signer efficiency, strong PQ security, and support for non-interactive authentication. These features make it ideal for IoT devices such as medical wearables or environmental trackers, where low energy consumption and minimal processing are critical, and delayed verification is acceptable. A limitation of LiteQS is its reliance on a resourceful verifier to handle encrypted key evaluations and log validation, making it less

suitable for scenarios requiring real-time or edge-level verification. Nonetheless, for heterogeneous IoT environments with clear asymmetry between signer and verifier capabilities, as depicted in Section III (Figure 1), LiteQS offers an effective and scalable solution.

II. PRELIMINARIES

This section presents the notation and acronyms in TABLE 3, followed by an overview of the cryptographic primitives that form the foundation of our proposed scheme.

TABLE 3: Acronyms and notations

Notation	Description
PQC	Post-Quantum Cryptography
ECC	Elliptic Curve Cryptography
FHE	Fully Homomorphic Encryption
HORS	Hash to Obtain Random Subset
PKO-SGN	Public Key Outsourced Signature
EU-CMA	Existential Unforgeability against Chosen Message Attack
IND-CPA	Indistinguishably under Chosen Plaintext Attack
ROM	Random Oracle Model
PRF	Pseudo-Random Function
PPT	Probabilistic Polynomial Time
OWF	One-Way Function
LWE	Learning With Error
sk/PK	Private/Public key
msk/MPK	Master private/public key
ID_i	User identity (e.g., MAC address)
N	Total number of users
j	Signer state
x_i	Variable of the user ID_i
x_i^j	Variable for the user ID_i with the state j
$x_i^{j,\ell}$	ℓ^{th} element of variable x_i^j for the user ID_i with the state j
$x \xleftarrow{\$} \mathcal{S}$	Random selection from a set $\mathcal{S}$
$\|x\|$	Bit length of variable x
$\|\|$	String concatenation
$\oplus$	Bitwise-XOR operation
$H : \{0,1\}^* \rightarrow \{0,1\}^\kappa$	Cryptographic hash function
$f : \{0,1\}^* \rightarrow \{0,1\}^\kappa$	One-way function
$x \leftarrow \texttt{PRF}(k, M)$	Accepts a key k and message M as input. It outputs x
$C \leftarrow E_k(m)$	Encrypts of message m under the key k. It outputs C
$\{0,1\} \leftarrow \texttt{CMP}(x, y)$	Equality comparison function of two (e.g., 64-bit) numerical values x and y

The Davies-Meyer scheme (DM) [65] is a generic and iterated cryptographic hash function based on a block cipher E. In LiteQS, we only rely on the one-wayness (OWF) of DM, which is based on the IND-CPA security of the symmetric cipher E. The DM algorithm is described as follows:

Definition 1: $B_n \leftarrow \texttt{DM}(M, B_0)$: Given a message M, a predefined initial value $I_{\texttt{DM}} = B_0$, and a block cipher E of length k, it splits M into n chunks $M = \{m_i\}_{i=1}^n$ where $n = \lceil \frac{|M|}{k} \rceil$, and computes $B_i = E_{m_i}(B_{i-1}) \oplus m_i, \forall i = 1, 2, ..., n$. Finally, it outputs B_n as the hash output.

Hash to Obtain Random Subset (HORS) [51] is an efficient hash-based one-time signature scheme that leverages the subset-resilience property of the underlying hash function and the one-wayness of the employed pseudorandom function (PRF). HORS is formally defined as follows:

Definition 2: HORS consists of three core algorithms, $\texttt{HORS} = (\texttt{Kg}, \texttt{Sig}, \texttt{Ver})$ described as follows:

- $(sk, PK, I_{\texttt{HORS}}) \leftarrow \texttt{HORS.Kg}(1^\kappa)$: Given the security parameter κ, it selects $I_{\texttt{HORS}} \leftarrow (k, t)$, generates t random κ-bit strings $\{s_i\}_{i=1}^t$, and computes $v_i \leftarrow f(s_i), \forall i = 1, \ldots, t$. It sets $sk \leftarrow \{s_i\}_{i=1}^t$ and $PK \leftarrow \{v_i\}_{i=1}^t$.
- $\sigma \leftarrow \texttt{HORS.Sig}(sk, M)$: Given sk and message M, it computes $h \leftarrow H(M)$. It splits h into k substrings $\{h_j\}_{j=1}^k$ (where $|h_j| = \log_2 t$) and interprets them as integers $\{i_j\}_{j=1}^k$. It outputs $\sigma \leftarrow \{s_{i_j}\}_{j=1}^k$.
- $b \leftarrow \texttt{HORS.Ver}(PK, M, \sigma)$: Given PK, M, and σ, it computes $\{i_j\}_{j=1}^k$ as in $\texttt{HORS.Sig}(.)$. If $v_{i_j} = f(\sigma_j), \forall j = 1, \ldots, k$, it returns $b = 1$, otherwise $b = 0$.

A Fully Homomorphic Encryption (FHE) scheme enables arbitrary computation on encrypted data without revealing the underlying plaintexts. Originating from Gentry's seminal work in 2009 [66], FHE allows an entity to perform functions over ciphertexts such that the result, once decrypted, matches the output of the same function applied to the plaintexts. An FHE scheme is defined as follows:

Definition 3: An FHE scheme [67] consists of four probabilistic polynomial-time algorithms $\texttt{FHE} = (\texttt{Kg}, \texttt{Enc}, \texttt{Eval}, \texttt{Dec})$ defined as below:

- $(sk', PK', I_{\texttt{FHE}}) \leftarrow \texttt{FHE.Kg}(1^\kappa)$: Given κ, it creates the auxiliary argument $I_{\texttt{FHE}}$ and generates FHE private/public key pair (sk', PK').
- $C \leftarrow \texttt{FHE.Enc}(PK', M)$: Given PK' and a plaintext M, it encrypts M and returns the ciphertext C.
- $C \leftarrow \texttt{FHE.Eval}(PK', \mathcal{F}(\overrightarrow{c} = \{c_j\}_{j=1}^n))$: Given PK', a function $\mathcal{F}$, and a set of input arguments $\overrightarrow{c}$, it evaluates $\mathcal{F}$ on $\overrightarrow{c}$ under encryption.
- $M \leftarrow \texttt{FHE.Dec}(sk', C)$: Given sk' and C, it decrypts C via sk' and outputs the plaintext M.

For illustration, $\texttt{FHE.Eval}(PK', \texttt{PRF}(Y, x))$ and $\texttt{FHE.Eval}(PK', \texttt{CMP}(X_1, X_2))$ evaluate $\texttt{PRF}(y, x)$ and $\texttt{CMP}(x_1, x_2)$ functions under encryption, where the key Y and the numerical values (X_1, X_2) are the encryption of y, x_1, and x_2 under PK' (i.e., $Y \leftarrow \texttt{FHE.Enc}(PK', y)$, $X_1 \leftarrow \texttt{FHE.Enc}(PK', x_1)$, $X_2 \leftarrow \texttt{FHE.Enc}(PK', x_2)$), respectively. We choose an IND-CPA-secure FHE instantiated with the Ring Learning With Error (R-LWE) variant of the BGV cryptosystem [68]. Note that these FHE instantiations also have a PQ security premise [69].

A Public Key Outsourced Signature scheme (PKO-SGN) transforms the one-time HORS signature scheme into a multiple-time hash-based digital signature. It leverages FHE to provide verifiers with one-time public keys without the knowledge of the private key. Moreover, PKO-SGN implements the one-way function (f) in HORS using the single-block-length DM construction.

Definition 4: A Public Key Outsourced Signature $\texttt{PKO-SGN} = (\texttt{Kg}, \texttt{Sig}, \texttt{PKConstr}, \texttt{Ver})$ is defined as follows:

- $(PK, sk, I) \leftarrow \texttt{PKO-SGN.Kg}(1^\kappa, \overrightarrow{ID})$: Given κ and a set of users' identifiers $\overrightarrow{ID}$, it returns PK with both FHE and master public keys $PK = \langle PK', MPK \rangle$, the private key $sk = \overrightarrow{\gamma}$, and the system-wide parameters $I \leftarrow (I_{\texttt{HORS}}, I_{\texttt{DM}}, I_{\texttt{FHE}})$.
- $\sigma_i^j \leftarrow \texttt{PKO-SGN.Sig}(\gamma_i, M_j)$: Given the seed $\gamma_i \in \overrightarrow{\gamma}$ of ID_i and a message M_j, it returns the signature σ_i^j.
- $cv_i^j \leftarrow \texttt{PKO-SGN.PKConstr}(PK, ID_i, j)$: Given the signer ID_i, state j, and PK, it constructs the required public keys under encryption cv_i^j via $\texttt{FHE.Eval}(.)$.
- $b \leftarrow \texttt{PKO-SGN.Ver}(PK_i^j, M_j, \sigma_i^j)$: Given PK_i^j, M_j, and σ_i^j, it outputs $b = 1$ if σ_i^j is valid, or $b = 0$ otherwise.

III. SYSTEM ARCHITECTURE AND SECURITY MODEL

System Model: We adopt the traditional public-key-based broadcast authentication model, tailored for diverse IoT-enabled applications while addressing key design considerations. Figure 1 illustrates the entities in our architecture, detailed as follows:

• *Signer:* A resource-constrained IoT device, such as a smart sensor, pacemaker, or implantable medical device (see Figure 1), responsible for signing and broadcasting messages to verifiers. These messages often contain sensitive data (e.g., heart rate, security logs), where data integrity and source authenticity are crucial for usability. The signer prioritizes efficient, energy-aware computations with minimal memory and bandwidth usage, while ensuring long-term security.

• *Verifier:* A resourceful entity (e.g., cloud server, physician, monitoring center) that authenticates messages from signers. It independently constructs one-time public keys from the master public key (MPK), enabling it to derive any public key PK_i^j for any user ID_i in a network of (e.g., $N = 2^{20}$) signers. Additionally, we propose an optional approach where non-resourceful verifiers can outsource public key derivation to a resourceful entity (e.g., a cloud server) via FHE computations, without relying on semi-honest entities or trusted parties.

Threat and Security Model: Our threat model is based on an Probabilistic Polynomial Time (PPT) adversary $\mathcal{A}$ equipped with the following capabilities:

1) *Passive attacks:* aim to monitor and interpret the output of the signature generation interface.
2) *Active attacks:* aim to intercept, forge, and modify messages and signatures sent from IoT devices. We assume that the adversary is equipped with a quantum computer.

We follow the standard Existential Unforgeability under Chosen Message Attack (EU-CMA) model [70]. The EU-CMA experiment for an PKO-SGN signature scheme is defined as follows:

Definition 5: The EU-CMA experiment $Expt_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}$ for an PKO-SGN scheme is defined as follows:

- $(PK, sk, I) \leftarrow \texttt{PKO-SGN.Kg}(1^\kappa, \overrightarrow{ID})$
- $(M^*, \sigma^*) \leftarrow \mathcal{A}^{\texttt{PKO-SGN.Sig}_{sk}(.), \texttt{PKO-SGN.PKConstr}(.)}(PK)$:
- If $1 \leftarrow \texttt{PKO-SGN.Ver}(PK, M^*, \sigma^*)$ and M^* was not queried to $\texttt{PKO-SGN.Sig}_{sk}(.)$, then return 1, else 0.

The advantage of $\mathcal{A}$ in this experiment is defined as $Adv_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}(\mathcal{A}) = Pr[Expt_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}} = 1]$. The EU-CMA advantage of PKO-SGN is defined as $Adv_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}(t, q_s) = \max\{Adv_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}(\mathcal{A})\}$, where t is the time complexity of $\mathcal{A}$ and q_s is the number of queries to the public key constructor and signing oracles. $\texttt{PKO-SGN.Sig}_{sk}(.)$ and $\texttt{PKO-SGN.PKConstr}(.)$ are as follows:

1) *Signing oracle* $\texttt{PKO-SGN.Sig}_{sk}(.)$: Given an input message M, it outputs a signature $\sigma \leftarrow \texttt{PKO-SGN.Sig}_{sk}(M)$.
2) Public key construct oracle $\texttt{PKO-SGN.PKConstr}(.)$: Given the public key PK, user identity ID_i, and counter j, it returns the one-time public key PK_i^j. Note that unlike previous public key constructors (e.g., [3], [11]), $\texttt{PKO-SGN.PKConstr}(.)$ does not require a root of trust on introduced entities (e.g., [3], [28]) and can be run based on public key data. $\texttt{PKO-SGN.PKConstr}(.)$ may be run by the verifier or a resourceful third party.

IV. THE PROPOSED SCHEME

We first present our proposed scheme, LiteQS. We then describe its instantiations, design rationale, and optimizations.

Design Motivation: To meet the criteria outlined in Section I, including PQ security, signer-side efficiency, minimal energy, memory, and bandwidth usage, and low cryptographic assumptions, we focus on optimizing the signer. Our design requires only a small, constant number of PRF calls per signing, resulting in significant energy savings. It uses a single 128-bit seed as the private key and transmits a compact 256-byte signature per message, making it the most lightweight among PQ counterparts (see performance evaluation VI). Additionally, it supports scalability by eliminating the need for certificates through a single master public key and enables efficient online-offline verification.

Comparative Justification: The main bottleneck of hash-based digital signatures is the generation and management of one-time public keys. As outlined in Section I-B, the existing alternatives rely on hyper-tree structures that incur extreme signature generation and transmission overhead. A trivial yet insecure approach would be to share the master secret key with a trusted party that replenishes one-time keys for the verifiers (e.g., [60]). However, this invalidates the non-repudiation and makes the system vulnerable to key compromises. Moreover, it is not scalable to large-IoTs due to the massive transmission overhead. Our proposed LiteQS framework directly addresses these challenges through a streamlined and secure mechanism, as detailed below.

Mechanistic Explanation: We address this public key management conundrum by introducing a novel approach that permits verifiers to construct one-time keys from a master public key via encrypted evaluations. Our idea is to wrap the master secret key with homomorphic encryption and then enable any verifier to retrieve one-time public keys for any valid signer ID_i and message M_i^j. This allows signers to achieve optimal efficiency concerning HORS since it only computes and broadcasts one HORS signature per message. The verifiers can construct one-time public keys via encrypted evaluations

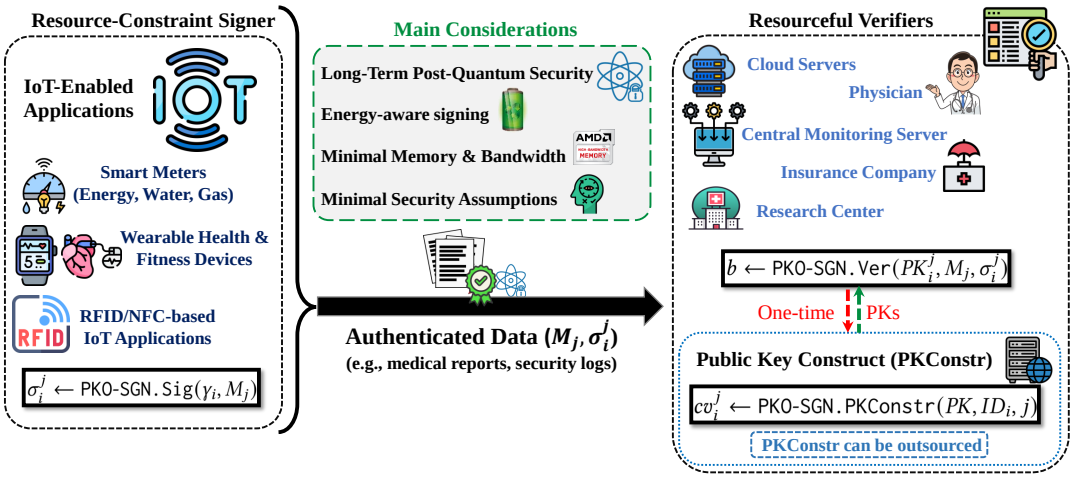

FIGURE 1: System model

without the risk of private key compromises. Our approach effectively transforms one-time HORS into practically unbounded hash-based signatures with minimal signer computations, and therefore, fittingly, we name our new scheme LiteQSign. The main focus of LiteQS is on heterogeneous IoT-enabled applications, where resource-limited devices perform signing or authentication, and a resourceful verifier handles public key derivation and verification. We provide the details of LiteQS in Algorithm 1.

Algorithmic Description: The key generation algorithm LiteQS.Kg, first derives the master signing key msk and sets up the public parameters I including HORS, FHE, and DM parameters as in Definition 2, 3, 1, respectively (Step 1). It derives the initial private key γ_i (seed) of each signer ID_i (Step 2-3). It then generates an FHE key pair (sk', PK'), encrypts msk with PK' to generate the master public key MPK, and sets LiteQS public key as $PK = (PK', MPK)$ (Step 6). As elaborated in public key construction, this permits any verifier to extract a one-time public key from the master public key under encryption without exposing it. Finally, all key pairs are distributed to the verifiers and signers (Step 7).

The signature generation LiteQS.Sig for signer ID_i begins by deriving the private key sk_i^j from the seed γ_i based on the message state (counter) j (Step 1). The signing process follows HORS.Sig, except that the signature elements $\{s_i^{j,\ell}\}_{\ell=1}^k$ are computed via PRF evaluations using sk_i^j instead of random generation (Steps 2-5). Finally, the signer updates the state j and discloses the HORS signature (Step 5).

LiteQS.PKConstr algorithm enables any verifier to generate the one-time public key PK_i^j under FHE encryption associated with a valid $ID_i \in \overrightarrow{ID}$ without any interaction with the signer or having to access private keys (msk, sk'). It first derives the initial seed γ_i of ID_i under FHE encryption that is preserved in $c\gamma_i$ (Step 1). It then pinpoints the private key sk_i^j of state j, which is sealed under csk_i^j (Step 2). Note that the signer uses sk_i^j to derive HORS signature components for M_i^j. Finally, it generates the FHE encryption of HORS one-time public key for the state j by evaluating $f(.)$ and PRF under encryption (Steps 3-5).

The signature verification LiteQS.Ver resembles HORS.Ver, but starts by constructing public keys using LiteQS.PKConstr and the signature verification is performed under encryption. The verifier performs f evaluations on the received k elements of the signature subset and encrypts the output using FHE. Next, the verifier evaluates the comparison function CMP under encryption via FHE.Eval. As we will shortly discuss in Section IV-A, the verifier may construct public keys offline before receiving the message-signature pair. Additionally, to reduce the computational demands, the verifier may use an alternative method by providing the indices (i.e., $\{x_i^{j,\ell}\}_{\ell=1}^k$ in Step 2, LiteQS.Sig) instead of the counter j to the LiteQS.PKConstr routine.

A. LiteQS INSTANTIATIONS AND OPTIMIZATIONS

The generic LiteQS in Algorithm 1 can be instantiated with any FHE, PRF and $f(.)$ as OWF. However, these instantiation choices make a drastic impact on performance, security, and practicality. In the following, we articulate our instantiation rationale and their potential optimizations.

BGV Cryptosystem as the FHE Instantiation: There exist various classes and schemes of FHE. We instantiated our FHE with BGV cryptosystem [68] for the following reasons: (i) BGV is considered as a benchmark for FHE instantiations. It is well-studied and implemented in different libraries like HElib[2]. (ii) We employ the Ring-Learning With Error (R-LWE) based BGV that offers an ideal security-efficiency trade-off. (iii) BGV is amenable to parallelism and supports CRT-based encoding techniques to allow entry-wise arithmetic. (iv) It facilitates leveled-FHE, enabling the evaluation of a predetermined depth circuit without necessitating any bootstrapping.

Performance Hurdles of Traditional Cryptographic Hash Functions in FHE Settings: Presuming it takes hundreds of clock cycles for a modern processor to handle a single block cipher encryption, it takes millions of clock cycles to complete the same task under FHE. Since LiteQS.PKConstr

[2]https://github.com/homenc/HElib

Algorithm 1 LiteQSign Scheme (LiteQS)

$(PK, \overrightarrow{\gamma}, I) \leftarrow$ LiteQS.Kg$(1^\kappa, \overrightarrow{ID} = \{ID_i\}_{i=1}^N)$:

1: $msk \xleftarrow{\$} \{0,1\}^\kappa$ and set $I \leftarrow (I_{\text{HORS}} = (k,t), I_{\text{FHE}}, I_{\text{DM}})$ according to Definitions 2, 3, 1.
2: **for** $i = 1, \ldots, N$ **do**
3: $\quad \gamma_i \leftarrow \text{PRF}(msk, ID_i)$
4: **end for**
5: $(sk', PK', I_{\text{FHE}}) \leftarrow \text{FHE.Kg}(1^\kappa)$
6: $MPK \leftarrow \text{FHE.Enc}(PK', msk)$, $PK = \langle PK', MPK \rangle$
7: **return** $(PK, \overrightarrow{\gamma} = \{\gamma_i\}_{i=1}^N, I)$, where γ_i is securely given to ID_i

$\sigma_i^j \leftarrow$ LiteQS.Sig(γ_i, M_i^j): The signer ID_i computes a signature on a message M_i^j as follows:

1: $sk_i^j \leftarrow \text{PRF}(\gamma_i, j)$
2: $h_i^j \leftarrow H(M_i^j)$, split h_i^j into k sub-strings $\{h_i^{j,\ell}\}_{\ell=1}^k$ where $|h_i^{j,\ell}| = \log_2 t$, and interpret each $\{h_i^{j,\ell}\}_{\ell=1}^k$ as an integer $\{x_i^{j,\ell}\}_{\ell=1}^k$.
3: **for** $\ell = 1, \ldots, k$ **do**
4: $\quad s_i^{j,\ell} \leftarrow \text{PRF}(sk_i^j, x_i^{j,\ell})$
5: **end for**
6: Set $j \leftarrow j+1$
7: **return** $\sigma_i^j = (s_i^{j,1}, s_i^{j,2}, ..., s_i^{j,k}, j)$

$cv_i^j \leftarrow$ LiteQS.PKConstr(PK, ID_i, j): Performed by the verifier for a given $ID_i \in \overrightarrow{ID}$ and state j, in *offline* mode before receiving signatures, or optionally outsourced to a powerful entity.

1: $c\gamma_i \leftarrow \text{FHE.Eval}(PK', \text{PRF}(MPK, ID_i))$
2: $csk_i^j \leftarrow \text{FHE.Eval}(PK', \text{PRF}(c\gamma_i, j))$
3: **for** $\ell = 1, \ldots, t$ **do**
4: $\quad cv_i^{j,\ell} \leftarrow \text{FHE.Eval}(PK', f(\text{PRF}(csk_i^j, \ell)))$
5: **end for**
6: **return** $cv_i^j \leftarrow (cv_i^{j,1}, cv_i^{j,2}, ..., cv_i^{j,t})$

$b_i^j \leftarrow$ LiteQS.Ver(PK, M_i^j, σ_i^j):

1: $cv_i^j \leftarrow \text{LiteQS.PKConstr}(PK, ID_i, j)$
2: Execute Step 2 in LiteQS.Sig
3: **for** $\ell = 1, \ldots, k$ **do**
4: $\quad v_i^{j,\ell} \leftarrow f(s_i^{j,\ell})$
5: $\quad CV_\ell^j \leftarrow \text{FHE.Enc}(PK', v_\ell^j)$
6: $\quad b_i^{j,\ell} \leftarrow \text{FHE.Eval}(PK', \text{CMP}(cv_i^{j,x_i^{j,\ell}}, CV_i^{j,\ell}))$
7: **end for**
8: **if** $b_i^{j,\ell} = 1, \forall \ell = 1, \ldots, k$ **then**, **return** $b_i^j = 1$ **else**, **return** $b_i^j = 0$
9: **end if**

requires FHE evaluations, we require FHE-friendly cryptographic primitives that suit the needs of LiteQS. The hash-based signatures usually rely on traditional hash functions H to realize both the message compression and one-way function $f(.)$. However, it was shown that ARX-based primitives like SHA-256 and BLAKE are not suitable for FHE evaluations. For instance, SHA-256 requires 3311 FHE levels, which is infeasible for many practical purposes [71]. Recent efforts have explored homomorphic evaluation of hash functions such as SHA256, SM3, etc., utilizing FHE schemes like TFHE [72] that enable rapid bootstrapping. However, they remain considerably distant from practical application, with execution times on the order of minutes [73], [74].

Mitigating Encrypted Evaluation Hurdles via Davies-Meyer as OWF*:* We made a key observation that $f(.)$ needs only OWF property but not a full cryptographic hash function. This permits us to consider alternative hash designs that rely on symmetric ciphers that are suitable for FHE evaluations. Consequently, we can leverage the best properties from both cryptographic realms.

The symmetric ciphers generally have lower multiplicative complexity (depth and size) compared to the traditional hash functions, with cheaper linear operations favoring more efficient FHE evaluations. Moreover, when evaluated under encryption, they can serve as OWF with proper instantiations. We have investigated various options and identified that a block cipher-based hash function named, Davies-Meyer (DM) [65], satisfies our efficiency and OWF prerequisites for the encrypted evaluation purposes. Compared to other constructions, DM structure is lighter than one-way double-block-length compression methods (e.g. Hirose [75]), and allows for key-setup and encryption parallelization as opposed to other single-block-length one-way compression functions.

Selection of Suitable Cipher for DM *Instantiaton:* We adopt AES as the underlying primitive for our DM instantiation based on several compelling factors. (i) AES is a widely adopted and standardized block cipher with numerous optimized implementations ranging from high-performance commodity platforms to resource-constrained embedded MCUs. (ii) It features a low number of rounds and avoids complex integer operations, making it suitable for constrained environments. (iii) The algebraic structure of AES is particularly amenable to parallel computation, batching via packing techniques, and hardware-level acceleration such as GPU-based processing [71]. (iv) In comparison to hash-based alternatives, an AES-based DM requires a smaller, fixed-size memory footprint for iteratively storing intermediate hash values, which is advantageous for memory-constrained deployments. (v) Finally, homomorphic evaluation of AES has been extensively studied and is supported by established libraries (e.g., HElib [76]), further affirming its suitability for our design.

Optimizations: To enhance the efficiency of signature verification, we incorporate a suite of online-offline optimizations that strategically decouple costly computations from time-sensitive operations. These optimizations shift the computational burden to the offline phase, thereby enabling faster verification during the online phase.

(i) The construction of the public key is independent of the message being verified and can be performed in advance for any identity ID_i and corresponding state information. This allows the verifier to execute LiteQS.PKConstr in batch mode offline and precompute the encrypted public keys. These precomputed values can then be efficiently leveraged for rapid online verification. As demonstrated in Section VI, this design yields substantial performance improvements for the online phase. *(ii)* Instead of computing the entire set of t one-time public keys, the verifier may selectively generate only the k components required for a given verification. This selective construction significantly reduces the number of

FHE evaluations required, thereby lowering both computation time and resource usage. *(iii)* Since LiteQS.PKConstr is publicly executable and does not require access to any private input, the verifier may optionally delegate its offline execution to a more resource-capable external entity, such as a cloud server. In return for tolerating minor transmission latency, the verifier offloads the intensive FHE evaluations, allowing the delegated server to exploit extensive parallelism and GPU acceleration. This is particularly beneficial in the context of our LiteQS instantiations, which are designed to take advantage of such computational enhancements.

V. SECURITY ANALYSIS

We prove that LiteQS is EU-CMA secure as follows.

Theorem 1: $Adv^{\text{EU-CMA}}_{\text{LiteQS}}(t,q_s) \leq q_s \cdot Adv^{\text{EU-CMA}}_{\text{HORS}}(t',q'_s)$, where $q'_s = q_s + 1$ and $\mathcal{O}(t') = \mathcal{O}(t) + q_s \cdot (k \cdot \text{PRF} + (t+2) \cdot \text{FHE.Eval(PRF)})$ (we omit terms negligible in terms of κ).

Proof: Let $\mathcal{A}$ be the LiteQS attacker. We construct a simulator $\mathcal{F}$ that uses $\mathcal{A}$ as a subroutine to break one-time EU-CMA secure HORS, where $(\overline{sk}, \overline{PK}, I_{\text{HORS}}) \leftarrow \text{HORS.Kg}(1^\kappa)$ (Definition 2). $\mathcal{F}$ is given the challenge $\overline{PK}$, on which $\mathcal{A}$ aims to produce a forgery. $\mathcal{F}$ has access to the HORS signing oracle under secret key $\overline{sk}$. $\mathcal{F}$ maintains two lists $\mathcal{LM}$ and $\mathcal{LS}$ to record the queried messages and $\text{LiteQS.Sig}_{sk}(.)$ outputs, respectively. $\mathcal{F}$ randomly chooses a target forgery index[3] $w \in [1,q_s]$. $\mathcal{A}$ uses a user identity $ID_i \in \overrightarrow{ID}$, where $i \xleftarrow{\$} \{1,\ldots,N\}$.

Algorithm $\mathcal{F}(\overline{PK}, I_{\text{HORS}})$

• *Setup:* $\mathcal{F}$ is run as in Definition 5:

(1) $msk \xleftarrow{\$} \{0,1\}^\kappa$.
(2) $I \leftarrow (I_{\text{HORS}}, I_{\text{FHE}}, I_{\text{DM}})$, where $(I_{\text{FHE}}, I_{\text{DM}})$ are as in Definition 3-1, respectively.
(3) $(sk', PK', I_{\text{FHE}}) \leftarrow \text{FHE.Kg}(1^\kappa)$.
(4) $MPK \leftarrow \text{FHE.Enc}(PK', msk)$; $PK = (PK', MPK)$.
(5) $sk_i^0 \leftarrow \text{PRF}(msk, ID_i)$.
(6) $sk = \{sk_i^j \leftarrow \text{PRF}(sk_i^0, j)\}_{j=1, j\neq w}^{q_s}$.
(7) $\{cv_i^j \leftarrow \text{LiteQS.PKConstr}(PK, ID_i, j)\}_{j=1,j\neq w}^{q_s}$.

Execute $\mathcal{A}^{\text{LiteQS.Sig}_{sk}(\cdot),\ \text{LiteQS.PKConstr}(\cdot),\ \text{HORS.Sig}_{\overline{sk}}(\cdot)}(PK, \overline{PK})$:

• *Queries:* $\mathcal{F}$ handles $\mathcal{A}$'s queries as follows:

(1) $\text{LiteQS.Sig}_{sk}(.)$*:* $\mathcal{F}$ returns $\sigma_i^w \leftarrow \text{HORS.Sig}_{\overline{sk}}(M_i^w)$ by querying HORS signing oracle, if $j = w$. Otherwise, $\mathcal{F}$ runs the steps (2-5) in LiteQS.Sig to compute σ_i^j under sk_i^j. $\mathcal{F}$ inserts M_i^j to $\mathcal{LM}$and (M_i^j, σ_i^j) to $\mathcal{LS}$as $\sigma_i^j \leftarrow \mathcal{LS}[M_i^j]$.

(2) $\text{LiteQS.PKConstr}(.)$ *Queries:* If $j = w$ then $\mathcal{F}$ returns $cv_i^w = \text{FHE.Enc}(PK', \overline{PK})$. Otherwise, $\mathcal{F}$ returns cv_i^j.

• *Forgery of* $\mathcal{A}$*:* $\mathcal{A}$ produces a forgery (M^*, σ^*) on $\overline{PK}$. $\mathcal{A}$ wins the EU-CMA experiment if $1 \leftarrow \text{LiteQS.Ver}(PK, M^*, \sigma^*)$ and $M^* \notin \mathcal{LM}$ conditions hold, and returns 1, else returns 0.

• *Forgery of* $\mathcal{F}$*:* If $\mathcal{A}$ fails to win the EU-CMA experiment for LiteQS, $\mathcal{F}$ also fails to win the EU-CMA experiment for HORS. As a result, $\mathcal{F}$ *aborts* and returns 0. Otherwise, $\mathcal{F}$ checks if $1 \leftarrow \text{HORS.Ver}(\overline{PK}, M^*, \sigma^*)$ and M^* was not queried to the HORS signing oracle (i.e., $\text{HORS.Sig}_{\overline{sk}}(.)$). If these conditions hold, $\mathcal{F}$ wins the EU-CMA experiment against HORS and returns 1. Otherwise, $\mathcal{F}$ *aborts* and returns 0.

• *Success Probability Analysis:* We analyze the events that are needed for $\mathcal{F}$ to win the EU-CMA experiment as follows:

(1) $\mathcal{F}$ *does not abort during* $\mathcal{A}$*'s queries with* $Pr[\overline{Abort1}]$: $\mathcal{F}$ can answer all of $\mathcal{A}$'s signature queries, since it knows all private keys except $j = w$, for which it can retrieve the answer from HORS signature oracle. $\mathcal{F}$ sets $PK_i^w = \text{FHE.Enc}(PK', \overline{PK})$ and can answer all other queries by running the public key construction algorithm. The only exception occurs if FHE.Eval(.) produces an incorrect PK_i^j during the simulation, which occurs with a negligible probability in terms of κ due to the correctness property of FHE. Therefore, we conclude $Pr[\overline{Abort\ 1}] \approx 1$.

(2) $\mathcal{A}$ *produces a valid forgery with* $Pr[Forge|\overline{Abort1}]$: If $\mathcal{F}$ does not abort during the queries, then $\mathcal{A}$ also does not abort, since its simulated view is computationally indistinguishable from the real view (see indistinguishability argument below). Hence, the probability that $\mathcal{A}$ produces a forgery against LiteQS is $Pr[Forge|\overline{Abort1}] = Adv^{\text{EU-CMA}}_{\text{LiteQS}}(q_s,t)$. There are three events that may also lead to $\mathcal{A}$'s forgery: (i) $\mathcal{A}$ breaks the subset-resiliency of H, whose probability is negligible in terms of κ [51]. (ii) $\mathcal{A}$ breaks IND-CPA secure FHE and recovers the master secret key msk, which permits a universal forgery. The probability that this happens is negligible in terms of κ for sufficiently large security parameters [68]. (iii) $\mathcal{A}$ breaks the evaluation of the comparison circuit for all k signatures (i.e., $b_i^{j,\ell} = 1, \forall \ell = 1,\ldots,k$), which occurs with a probability that is $\frac{1}{k}\times$ negligible in relation to κ. (iv) $\mathcal{A}$ inverts DM by breaking the underlying IND-CPA cipher, which also happens with negligible probability in terms of κ [65]. Therefore, they are omitted in the theorem statement.

(3) $\mathcal{F}$ *does not abort after* $\mathcal{A}$ *'s forgery with* $Pr[\overline{Abort2}|\overline{Abort1} \wedge Forge]$: $\mathcal{F}$ does not abort if $\mathcal{A}$'s forgery is on the target public key PK_i^w. Since w is randomly selected from $[1,q_s]$, this occurs with $1/q_s$.

(4) $\mathcal{F}$ *wins the* EU-CMA *experiment with* $Adv^{\text{EU-CMA}}_{\text{HORS}}(t',q'_s)$: $Pr[Win] = Pr[\overline{Abort1}] \cdot Pr[Forge|\overline{Abort1}] \cdot Pr[\overline{Abort2}|\overline{Abort1} \wedge Forge]$. Therefore, $Pr[Win]$ is bounded as:

$$Adv^{\text{EU-CMA}}_{\text{LiteQS}}(t,q_s) \leq q_s \cdot Adv^{\text{EU-CMA}}_{\text{HORS}}(t',q'_s)$$

• *Execution Time Analysis:* The running time of $\mathcal{F}$ is that of $\mathcal{A}$ plus the time required to respond to q_s public key and signature queries. Each signature query demands H and $k \cdot \text{PRF}(.)$; and each LiteQS.PKConstr(.) query needs $(t+2)\cdot\text{FHE.Eval(PRF)}$. The approximate running time of $\mathcal{F}$ is $\mathcal{O}(t') = \mathcal{O}(t) + q_s \cdot (k \cdot \text{PRF} + (t+2)\text{FHE.Eval(PRF)})$.

[3]We follow SPHNICS+ [45] where the maximum number of signing queries is $2^{40} \leq q_s \leq 2^{60} \ll 2^\kappa$

• *Indistinguishability Argument*: In the real view of $\mathcal{A}$ ($\mathcal{A}_{real}$), all values are computed from the master secret key and seeds as in the key generation, signing, and public key construction algorithms. The simulated view of $\mathcal{A}$ ($\mathcal{A}_{sim}$) is identical to $\mathcal{A}_{real}$, except PK_i^w is replaced with the challenge HORS public key. This implies that $(sk_i^w = \overline{sk}, PK_i^w = \overline{PK})$ holds. Since HORS.Kg(.) generates the secret keys random uniformly (Definition 2), the joint probability distribution of (sk_i^w, PK_i^w) in $\mathcal{A}_{sim}$ is similar to that of $\mathcal{A}_{real}$. Therefore, $\mathcal{A}_{real}$ and $\mathcal{A}_{sim}$ are computationally indistinguishable. ■

Corollary 1: The LiteQS scheme provides PQ promises.
Proof: Based on our preceding formal security analysis and the incorporation of cryptographic primitives such as FHE, PRF, and hash functions, the LiteQS scheme ensures PQ assurances. Specifically, the PRF and hash functions, being symmetric cryptography primitives, remain unaffected by Shor's algorithm, while the impact of Grover's probabilistic algorithm can be mitigated by scaling up the sizes, considering the potential of quantum computers. Additionally, the FHE schemes, exemplified by our instantiation, the BGV scheme [68], are constructed upon lattice-based hard problems (e.g., General-LWE), which provide PQ security. ■

VI. PERFORMANCE ANALYSIS AND COMPARISON

In this section, we give a detailed performance analysis of LiteQS and compare with its counterparts.

A. EVALUATION METRICS AND EXPERIMENTAL SETUP

Evaluation Metrics: Given that LiteQS targets resource-limited IoT devices with a resourceful verifier, our analysis focuses on evaluating LiteQS and its comparable schemes, with particular emphasis on signer-side efficiency, including: **(i)** private key and signature sizes, which translate into a small memory footprint and low memory access requirements. This not only reduces the energy consumption but also frees up more memory for main applications. It is particularly important for low-end IoT devices, which are characterized by limited memory space and relatively expensive memory access (e.g., 8-bit AVR microcontrollers). **(ii)** signing computational efficiency which translates into reduced energy consumption and longer battery lifetime for resource-limited devices. **(iii)** long-term security (i.e., PQ security) in order to offer resiliency against the quantum computing breaches (e.g., Shor's algorithm [23]).

Parameter Selection: Our system-wide parameters are $I = (I_{\text{HORS}}, I_{\text{FHE}}, I_{\text{DM}})$. We choose $I_{\text{HORS}} \leftarrow (k = 16, t = 1024)$, where SHA-256 and DM are used as H and f (i.e., OWF), respectively. In I_{DM}, we choose AES-128 as our PRF. In I_{FHE}, we set the plaintext space of mod 2, the lattice dimension $\phi(m) = 46080$, where the m-th cyclotomic is $m = 53261$. We utilize a packing technique to evaluate 120 blocks of AES at once. We set $N = 2^{20}$ as the number of resource-constrained signers within the IoT network.

Hardware Configuration: We tested LiteQS on both commodity hardware and a low-end MCU: (i) *Commodity Hardware:* is a resourceful desktop equipped with Intel i9-9900K@3.6GHz processor and 64GB of RAM. (ii) *Embedded device:* We evaluate LiteQS on an 8-bit ATxmega128A1 microcontroller to assess its efficiency on embedded IoT devices. The MCU features 128 KB flash memory, 2 KB RAM, 8 KB EEPROM, and operates at a 32 MHz clock frequency.

Software Configuration: For the commodity hardware, we utilized the following libraries (i) OpenSSL[4] to implement SHA-256. (ii) HElib[5] to implement FHE functionalities (*e.g.*, evaluation and comparison under encryption[6]). (iii) DM is implemented using the hardware-optimized AES-NI [77]. For the 8-bit AVR device, we employed the AVR cryptographic library[7] to implement AES-128, offering an optimized assembly implementation, resulting in mimimal cycles to evaluate hashing and PRF calls.

Selection Rationale of Counterparts: The selection of our counterparts is based on the discussed evaluation metrics and the availability of open-source implementation and/or open-access benchmarks. Numerous digital signatures have been proposed in the literature that address the resource limitations of IoT devices. Nevertheless, few schemes address low-end embedded devices, such as our target 8-bit AVR MCU. In order to cover different signatures with the knowingly existing post-quantum intractability assumptions, we carefully selected the following constructions:

(i) Lattice-based: the NIST PQC standards ML-DSA-II [59] and FN-DSA-512 [36]. They are considered the most prominent lattice-based signatures, with balanced efficiency between key sizes and signing efficiency. We also selected BLISS-I because it is the only lattice-based signature with a benchmark on an 8-bit AVR MCU [78].

(ii) Hash-based: generally suffer from an expensive signing cost with larger key sizes. We selected the NIST PQC standard SLH-DSA [45], a stateless signature scheme. We also selected XMSS$^{\text{MT}}$ [46] as a standard stateful hash-based signature. To our knowledge, there is no hash-based signature with a benchmark on 8-bit AVR MCUs.

(iii) Conventional signatures: We also considered non-PQ signature schemes. Although they do not achieve long-term security, ECC-based signature schemes are signer-efficient with small key sizes. We selected the mostly-used standards, ECDSA [41] and Ed25519 [42]. Other conventional (e.g., pairing-based [79]) digital signatures incur expensive operations during signature generation, therefore, not practical for resource-limited IoT devices.

B. PERFORMANCE ON SIGNER

Performance comparisons on commodity hardware and the embedded device are shown in TABLEs 2 and 4, respectively.

• *Memory Usage:* LiteQS achieves the lowest memory

[4]https://github.com/openssl/openssl
[5]https://github.com/homenc/HElib
[6]https://github.com/iliailia/comparison-circuit-over-fq/tree/master
[7]https://github.com/cantora/avr-crypto-lib

usage by having the smallest private key size among its counterparts. For example, the private key of LiteQS is 3× and 114× smaller than that of the conventional signer-efficient Ed25519 and PQ-secure ML-DSA standards, respectively. The private key is 22× smaller than that of the most efficient lattice-based counterpart, BLISS-I [37], respectively. It is without incurring large code size and expensive costly sampling operations that may result in side-channel attacks [39]. Notably, LiteQS consumes minimal memory of only 2.8%. We argue that the cryptographic data should occupy minimal space, particularly in resource-limited devices (e.g., pacemakers). Indeed, the embedded devices generate system-related (e.g., log files) and application-related (e.g., sensory information) data, which may cause memory overflow, considering an additional high cryptographic memory usage.

• *Bandwidth Overhead:* LiteQS boasts a compact signature size that is 9.4× and 2.6× smaller than the NIST PQC standards, ML-DSA-II and Falcon-512, respectively. The signature size of LiteQS is also 22× smaller than that of the most-efficient lattice-based BLISS-I. A small signature size results in low transmission overhead, thereby minimizing energy consumption on resource-constrained IoT devices. This reduced energy expenditure is crucial for extending the operational lifespan of devices that often operate on limited power sources.

• *Signature Generation:* TABLE 2 demonstrates that among our counterparts (i.e., conventional-secure and post-quantum), LiteQS exhibits the fastest signing time and the lowest signer storage overhead. It is 10× and 43× faster than the NIST PQC standards, ML-DSA-II and Falcon-512, respectively. The computational performance advantages at the signer of LiteQS become even more apparent on embedded devices. Based on 8-bit AVR MCU results in TABLE 4, the signing time of LiteQS is 20× and 44× faster than the most efficient PQ-secure BLISS-I and conventional-secure Ed25519, respectively.

• *Energy Consumption:* The high signing efficiency translates into better energy awareness on low-end IoT devices. To demonstrate the potential of LiteQS, in TABLE 4 presents a comprehensive energy analysis that measures the energy consumption per signature generation and transmission. We follow [80], which considers a MICaz sensor node operating on an ATmega128L MCU, equipped with a ZigBee 2.4GHz radio chip (CC2420), and powered by an AA battery with an energy capacity of 6750J. The sensor node drains 4.07nJ per cycle and 0.168μJ per bit transmission. We measure the energy consumption during one signature generation and transmission. We then measure the expected operation time of the IoT device based on different signing frequencies and assuming that the device performs only signature generation operations. Our findings reveal that LiteQS can operate with a signature generation and transmission frequency equal to 10 seconds and one minute for up to 5.95 and 35.7 years without battery replacement, respectively, while it is 95 days and 1.56 years for our sole PQ BLISS PQ counterpart, which is also not backward compatible with currently deployed cryptographic primitives. Moreover, LiteQS performs better than the conventional EC standards (ECDSA,Ed25519) by drawing 4.44× and 3.36× less energy capacity, respectively.

Remember that, to the best of our knowledge, none of the selected NIST PQC signature standards have an open-source implementation available on resource-limited devices (i.e., 8-bit microcontrollers). One of the most prominent PQ alternatives with a reported performance on this platform is BLISS-I [37]. We also included the most efficient ECC-based alternative Ed25519 and the widely-used ECDSA in our energy comparison to assess LiteQS performance with respect to (pre-quantum) conventional schemes. Our energy analysis showcases that LiteQS offer the longest battery lifetime when only the cryptographic computation is considered. Hence, we confirm that LiteQS is the most suitable signature scheme for highly resource-constrained IoT devices.

We note that BLISS-I is vulnerable to side-channel attacks, which hinders its use in practice. Side-channel attack resiliency and ease of implementation are important factors for the practical deployment of signature schemes on embedded devices. Lattice-based signatures require various types of sampling operations (e.g., Gaussian, rejection samplings) that make them vulnerable to side-channel attacks [38]. Moreover, due to their complexity, they are notoriously difficult to implement on such platforms. As an example, Falcon needs 53 bits of precision to implement without emulation [81], which hinders its deployment on 8-bit microcontrollers. LiteQS signature generation requires only a few PRF calls. Hence, it is free from the aforementioned specialized side-channel and timing attacks that target sampling operations. Moreover, it is easy to implement since it only requires a suitable symmetric cipher (e.g., AES) and a cryptographic hash function (e.g., SHA256) with a minimal code size. Our analysis validates that LiteQS is the most suitable alternative among its counterparts to be deployed for signing on IoT applications due to its high computational efficiency, compact key and signature sizes, and high security.

C. PERFORMANCE ON VERIFIER

While LiteQS is a signer-optimal scheme, we also introduced strategies to minimize the verifier's computational and storage overhead. As explained in Section IV-A, the verifiers can generate public keys in offline mode (before signature verification), thereby improving the efficiency of online verification. Moreover, the verifiers have the option to outsource offline public key construction to a resourceful entity.

Online Verification: The online verification cost is comprised of $k \times$ PRF(.), $k \times$ FHE.Enc(.), and $k \times$ FHE.Eval(.) of the comparison circuit. According to our implementation parameters, this is estimated to be approximately 1.913 seconds. Also, for further cost reduction, we strongly recommend an offline generation of public keys whenever possible.

In our instantiation, instead of generating a full set of t keys, the verifier can only construct k one-time public key components. Specifically, the verifier performs $k \times$ PRF(.)

TABLE 4: Performance comparison of LiteQS and counterparts at the signer on 8-bit AVR AtMega128A1 MCU

Scheme	Signing Overhead		Transmission Overhead		Total Energy (mJ)	Expected Operation w.r.t. Transmission Frequency				$\|sk\|$ (KB)	PQ	EoI	BC	SCA
	Time (cycles)	Energy (mJ)	Size (KB)	Energy (mJ)		f=1 sec	f=2 sec	f=10 sec	f=1 min					
ECDSA [41]	34,903,000	1.52	0.06	0.08	1.60	48.85d	97.71d	1.34y	8.01y	0.06	×	×	✓	✓
Ed25519 [42]	22,688,583	1.13	0.06	0.08	1.21	64.55d	129.10d	1.77y	10.65y	0.06	×	×	✓	✓
BLISS-I [37]	10,537,981	0.49	5.6	7.70	8.19	9.54d	19.08d	95.42d	1.56y	2	✓	×	×	✓
LiteQSign	**514,788**	**0.02**	**0.25**	**0.34**	**0.36**	**217.01d**	**1.19y**	**5.95y**	**35.68y**	**0.02**	✓	✓	✓	×

The counterpart selection covers the most efficient existing conventional (ECDSA, Ed25519) and PQ-secure (BLISS), with an available benchmark on the selected 8-bit AVR MCU. PQ denotes post-quantum security. EoI denotes ease of implementation. BC denotes backward compatibility. SCA denotes side-channel attacks found in the literature.

for the k signature elements, $k \times$ FHE.Enc(.) to obtain the encrypted version of them, and $k \times$ FHE.Eval(.) for evaluating the comparison circuit under encryption.

Offline Public Key Construction: The principal computational overhead in LiteQS arises from its offline phase. Empirical evaluations indicate that a single homomorphic AES evaluation per block requires approximately 2.29 seconds, resulting in a cumulative cost of 41.22 seconds to generate k public key components. This offline cost, however, can be substantially mitigated through parallelization strategies. Specifically, each public key element in the HORS construction can be independently computed, making it well-suited for distribution across multiple threads or processing units. Additionally, as detailed in Section IV-A, the BGV homomorphic encryption scheme supports extensive parallelization, which was a decisive factor in selecting AES as DM building block.

Several FHE libraries support BGV and offer hardware-level optimizations. For instance, Microsoft SEAL[8] and PALISADE[9] provide support for Advanced Vector Extensions (AVX), while OpenFHE[10] extends this capability to include GPU acceleration. Empirical benchmarks demonstrate that OpenFHE achieves a speedup of approximately 13× for multiplicative depth one, and up to 26× for depth five, compared to non-accelerated baselines [82].

Further efficiency gains can be achieved by leveraging the Chinese Remainder Theorem (CRT) [83], which facilitates the encryption of element-wise vectors, thereby enabling component-wise homomorphic operations such as additions and multiplications. This batching mechanism allows thousands of parallel function evaluations, such as AES instances, across distinct inputs. Notably, HElib has been shown in some configurations to outperform other libraries like Microsoft SEAL in batching efficiency, which is particularly relevant to our design. We intend to explore this optimization avenue in future work.

Verifier Storage Overhead: The total size of the master public key PK with the expansion per block evaluation is around 9.42 MB. If only a single signer is considered, the size of PK is much larger than that of its counterparts. However, it LiteQS enables a verifier to construct public keys for *any valid* signer ID_i of any state j. This unique property permits LiteQS to achieve compact storage for a large number of signers since the verifier does not need to store a certificate for their public keys. For example, the total storage (public key plus certificate) for 2^{20} users is still 9.42 MB for LiteQS, while it is around 1.52 GB and 3.74 GB for Falcon-512 and ML-DSA-II, respectively. The total storage advantage increases with a growing number of signers.

[8]Microsoft SEAL Library https://github.com/microsoft/SEAL

[9]PALISADE Library https://gitlab.com/palisade/palisade-release

[10]OpenFHE Library https://openfhe.org/

VII. CONCLUSION

This work addressed the critical limitations of deploying PQ digital signatures in resource-constrained IoT environments, where signer-side efficiency is vital for energy preservation, device longevity, and reliable real-time operation. While NIST-standardized PQ signature schemes provide strong quantum security, their substantial computational, memory, and communication overhead render them impractical for low-end IoT devices such as wearables, sensors, and embedded systems. To overcome these challenges, we proposed *LightQSign* (LiteQS), a novel lightweight PQ signature scheme that achieves near-optimal signature generation through a constant number of hash operations per signing and supports non-interactive, on-demand public key reconstruction by verifiers. LiteQS requires no trusted third parties, secure enclaves, or stateful key management, making it particularly suited for scalable and stateless deployment. We formally proved its security in the random oracle model and demonstrated its practical viability through comprehensive evaluations on both commodity and highly resource-constrained 8-bit AtMega128A1 microcontrollers. Our results show that LiteQS outperforms NIST PQC standards in signer efficiency, memory footprint, and energy consumption, offering a practical, quantum-resilient authentication solution for heterogeneous IoT ecosystems. As future work, we aim to extend this architecture to support more advanced security features and explore the use of optimized and parallelized FHE techniques with GPU acceleration to reduce verifier-side overhead, further enhancing the scalability of quantum-secure systems.

REFERENCES

[1] A. Ahad, M. Tahir, M. Aman Sheikh, K. I. Ahmed, A. Mughees, and A. Numani, "Technologies trend towards 5G network for smart health-care using iot: A review," *Sensors*, vol. 20, no. 14, p. 4047, 2020.

[2] M. Pradhan and J. Noll, "Security, privacy, and dependability evaluation in verification and validation life cycles for military iot systems," *IEEE Communications Magazine*, vol. 58, no. 8, pp. 14–20, 2020.

[3] R. Behnia and A. A. Yavuz, "Towards practical post-quantum signatures for resource-limited internet of things," in *Annual Computer Security Applications Conference*, 2021, pp. 119–130.

[4] C. Camara, P. Peris-Lopez, J. M. De Fuentes, and S. Marchal, "Access

control for implantable medical devices," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 3, pp. 1126–1138, 2020.
[5] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, D. Niyato, O. Dobre, and H. V. Poor, "6G internet of things: A comprehensive survey," *IEEE Internet of Things Journal*, 2021.
[6] Y.-H. Joung, "Development of implantable medical devices: from an engineering perspective," *International neurourology journal*, vol. 17, no. 3, p. 98, 2013.
[7] J. Sametinger, J. Rozenblit, R. Lysecky, and P. Ott, "Security challenges for medical devices," *Communications of the ACM*, vol. 58, no. 4, 2015.
[8] A. Mudgerikar and E. Bertino, "Iot attacks and malware," *Cyber Security Meets Machine Learning*, pp. 1–25, 2021.
[9] H. Kim, E. Kang, D. Broman, and E. A. Lee, "Resilient authentication and authorization for the internet of things (iot) using edge computing," *ACM Transactions on Internet of Things*, vol. 1, no. 1, pp. 1–27, 2020.
[10] I. Simsek, "Authentication, authorization, access control, and key exchange in internet of things," *ACM Transactions on Internet of Things*, vol. 5, no. 2, pp. 1–30, 2024.
[11] S. E. Nouma, , and A. A. Yavuz, "Post-quantum forward-secure signatures with hardware-support for internet of things," ser. IEEE International Conference on Communications (ICC). IEEE, 2023.
[12] D. Sisodia, J. Li, S. Mergendahl, and H. Cam, "A two-mode, adaptive security framework for smart home security applications," *ACM Transactions on Internet of Things*, vol. 5, no. 2, pp. 1–31, 2024.
[13] A. Goudarzi, F. Ghayoor, M. Waseem, S. Fahad, and I. Traore, "A survey on iot-enabled smart grids: emerging, applications, challenges, and outlook," *Energies*, vol. 15, no. 19, p. 6984, 2022.
[14] X. Ji, C. Li, X. Zhou, J. Zhang, Y. Zhang, and W. Xu, "Authenticating smart home devices via home limited channels," *ACM Transactions on Internet of Things*, vol. 1, no. 4, pp. 1–24, 2020.
[15] A. A. Yavuz and P. Ning, "Self-sustaining, efficient and forward-secure cryptographic constructions for unattended wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1204–1220, 2012.
[16] P. B. Adamson, "Pathophysiology of the transition from chronic compensated and acute decompensated heart failure: new insights from continuous monitoring devices," *Current heart failure reports*, vol. 6, no. 4, 2009.
[17] M. R. Zile, T. D. Bennett, M. St. John Sutton, Y. K. Cho, P. B. Adamson, and M. F. Aaron, "Transition from chronic compensated to acute decompensated heart failure: pathophysiological insights obtained from continuous monitoring of intracardiac pressures," *Circulation*, vol. 118, no. 14, 2008.
[18] W. Li, T. He, N. Jing, and L. Wang, "Mmhsv: In-air handwritten signature verification via millimeter-wave radar," *ACM Transactions on Internet of Things*, vol. 4, no. 4, pp. 1–22, 2023.
[19] A. Vangala, A. K. Das, V. Chamola, V. Korotaev, and J. J. Rodrigues, "Security in iot-enabled smart agriculture: Architecture, security solutions and challenges," *Cluster Computing*, vol. 26, no. 2, pp. 879–902, 2023.
[20] M. R. Rieback and B. Crispo, "Rfid guardian: A battery-powered mobile device for rfid privacy management," in *Australasian Conference on Information Security and Privacy*. Springer, 2005, pp. 184–194.
[21] C. Shi, J. Liu, H. Liu, and Y. Chen, "Wifi-enabled user authentication through deep learning in daily activities," *ACM Transactions on Internet of Things*, vol. 2, no. 2, pp. 1–25, 2021.
[22] H. Zhang, M. Babar, M. U. Tariq, M. A. Jan, V. G. Menon, and X. Li, "Safecity: Toward safe and secured data management design for iot-enabled smart city planning," *IEEE Access*, pp. 145 256–145 267, 2020.
[23] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.
[24] J.-P. Aumasson, "The impact of quantum computing on cryptography," *Computer Fraud & Security*, vol. 2017, no. 6, pp. 8–11, 2017.
[25] S. Darzi, K. Ahmadi, S. Aghapour, A. A. Yavuz, and M. M. Kermani, "Envisioning the future of cyber security in post-quantum era: A survey on pq standardization, applications, challenges and opportunities," *arXiv preprint arXiv:2310.12037*, 2023.
[26] C. Cheng, R. Lu, A. Petzoldt, and T. Takagi, "Securing the internet of things in a quantum world," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 116–120, 2017.
[27] M. Adeli, N. Bagheri, H. R. Maimani, S. Kumari, and J. J. Rodrigues, "A post-quantum compliant authentication scheme for iot healthcare systems," *IEEE Internet of Things Journal*, 2023.
[28] S. E. Nouma and A. A. Yavuz, "Trustworthy and efficient digital twins in post-quantum era with hybrid hardware-assisted signatures," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 20, no. 6, pp. 1–30, 2024.
[29] C. Camara, P. Peris-Lopez, and J. E. Tapiador, "Security and privacy issues in implantable medical devices: A comprehensive survey," *Journal of biomedical informatics*, vol. 55, pp. 272–289, 2015.
[30] A. L. Martínez, M. G. Pérez, and A. Ruiz-Martínez, "A comprehensive model for securing sensitive patient data in a clinical scenario," *IEEE Access*, vol. 11, pp. 137 083–137 098, 2023.
[31] S. E. Nouma and A. A. Yavuz, "Practical cryptographic forensic tools for lightweight internet of things and cold storage systems," in *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation*, 2023, pp. 340–353.
[32] K.-A. Shim, C.-M. Park, N. Koo, and H. Seo, "A high-speed public-key signature scheme for 8-b iot-constrained devices," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3663–3677, 2020.
[33] C. Costello and P. Longa, "SchnorrQ: Schnorr signatures on fourQ," *MSR Tech Report, 2016*, 2016.
[34] T. Dang, J. Lichtinger, Y.-K. Liu, C. Miller, D. Moody, R. Peralta, R. Perlner, A. Robinson *et al.*, "Module-lattice-based digital signature standard," *NIST, Thinh Dang, Jacob*, 2024.
[35] A. Sehgal, V. Perelman, S. Kuryla, and J. Schonwalder, "Management of resource constrained devices in the internet of things," *IEEE Communications Magazine*, vol. 50, no. 12, pp. 144–149, 2012.
[36] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "Falcon: Fast-fourier lattice-based compact signatures over NTRU," *Submission to the NIST's post-quantum cryptography standardization process*, 2018.
[37] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky, "Lattice signatures and bimodal gaussians," in *Cryptology Conf.*, 2013.
[38] E. Karabulut and A. Aysu, "Falcon down: Breaking falcon post-quantum signature scheme through side-channel attacks," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 691–696.
[39] M. Tibouchi and A. Wallet, "One bit is all it takes: a devastating timing attack on BLISS's non-constant time sign flips," *Journal of Mathematical Cryptology*, vol. 15, no. 1, pp. 131–142, 2021.
[40] T. Pornin, "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)," RFC 6979, Aug. 2013.
[41] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *International journal of information security*, vol. 1, no. 1, pp. 36–63, 2001.
[42] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of cryptographic engineering*, vol. 2, no. 2, pp. 77–89, 2012.
[43] G. Zhang, Y. Liao, Y. Fan, and Y. Liang, "Security analysis of an identity-based signature from factorization problem," *IEEE Access*, vol. 8, pp. 23 277–23 283, 2020.
[44] H. Liu, D. Han, M. Cui, K.-C. Li, A. Souri, and M. Shojafar, "Idenmultisig: Identity-based decentralized multi-signature in internet of things," *IEEE Transactions on Computational Social Systems*, 2023.
[45] D. Cooper *et al.*, "Stateless hash-based digital signature standard," 2024.
[46] A. Hülsing, L. Rausch, and J. Buchmann, "Optimal parameters for XMSSˆ MT," *Cryptology ePrint Archive, Paper 2017/966*, 2017.
[47] W. Beullens, "Mayo: practical post-quantum signatures from oil-and-vinegar maps," in *International Conference on Selected Areas in Cryptography*. Springer, 2021, pp. 355–376.
[48] G. K. Verma, B. Singh, N. Kumar, and V. Chamola, "Cb-cas: Certificate-based efficient signature scheme with compact aggregation for industrial internet of things environment," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2563–2572, 2019.
[49] X. Chen, D. He, M. K. Khan, M. Luo, and C. Peng, "A secure certificateless signcryption scheme without pairing for internet of medical things," *IEEE Internet of Things Journal*, vol. 10, pp. 9136–9147, 2022.
[50] NIST, "Post-Quantum Cryptography Standardization," https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization, accessed: May 23, 2024.
[51] L. Reyzin and N. Reyzin, "Better than BiBa: Short one-time signatures with fast signing and verifying," in *Australasian Conference on Information Security and Privacy*, 2002, pp. 144–153.
[52] A. Hülsing, L. Rausch, and J. Buchmann, "Optimal parameters for XMSS MT," in *International conference on availability, reliability, and security*, 2013, pp. 194–208.
[53] D. McGrew, M. Curcio, and S. Fluhrer, "Leighton-Micali Hash-Based Signatures," RFC 8554, Apr. 2019.
[54] R. C. Merkle, "A certified digital signature," in *Conference on the Theory and Application of Cryptology*. Springer, 1989, pp. 218–238.

[55] A. A. Yavuz, K. Sedghighadikolaei, S. Darzi, and S. E. Nouma, "Beyond basic trust: Envisioning the future of nextgen networked systems and digital signatures," in *2023 5th IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE Computer Society, 2023, pp. 267–276.

[56] M. Baldi, S. Bitzer, A. Pavoni, P. Santini, A. Wachter-Zeh, and V. Weger, "Zero knowledge protocols and signatures from the restricted syndrome decoding problem," in *IACR International Conference on Public-Key Cryptography*. Springer, 2024, pp. 243–274.

[57] S. Kim, J. Ha, M. Son, B. Lee, D. Moon, J. Lee, S. Lee, J. Kwon, J. Cho, H. Yoon *et al.*, "Aim: symmetric primitive for shorter signatures with stronger security," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 401–415.

[58] Y. Hashimoto, T. Takagi, and K. Sakurai, "General fault attacks on multivariate public key cryptosystems," in *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29–December 2, 2011. Proceedings 4*. Springer, 2011, pp. 1–18.

[59] G. Alagic, M. Bros, P. Ciadoux, D. Cooper, Q. Dang, T. Dang, J. Kelsey, J. Lichtinger, Y.-K. Liu, C. Miller *et al.*, "Status report on the first round of the additional digital signature schemes for the nist post-quantum cryptography standardization process," *NIST IR*, vol. 8528, 2024.

[60] W. Ouyang, Q. Wang, W. Wang, J. Lin, and Y. He, "Scb: Flexible and efficient asymmetric computations utilizing symmetric cryptosystems implemented with intel sgx," in *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. IEEE, 2021.

[61] K. Sedghighadikolaei and A. A. Yavuz, "A comprehensive survey of threshold digital signatures: Nist standards, post-quantum cryptography, exotic techniques, and real-world applications," *arXiv preprint arXiv:2311.05514*, 2023.

[62] S. E. Nouma and A. A. Yavuz, "Lightweight and high-throughput secure logging for internet of things and cold cloud continuum," *arXiv preprint arXiv:2506.08781*, 2025.

[63] A. Ahmad, S. Lee, and M. Peinado, "Hardlog: Practical tamper-proof system auditing using a novel audit device," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1791–1807.

[64] Z. Li, Q. A. Chen, R. Yang, Y. Chen, and W. Ruan, "Threat detection and investigation with system-level provenance graphs: A survey," *Computers & Security*, vol. 106, p. 102282, 2021.

[65] B. Preneel, "Davies-meyer hash function," in *Encyclopedia of Cryptography and Security*, 2005, pp. 136–136.

[66] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 169–178.

[67] F. Armknecht, C. Boyd, C. Carr, K. Gjøsteen, A. Jäschke, and Reuter, "A guide to fully homomorphic encryption," *Cryptology Archive*, 2015.

[68] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Transactions on Computation Theory (TOCT)*, vol. 6, no. 3, pp. 1–36, 2014.

[69] Y. Yu and X. Xie, "Privacy-preserving computation in the post-quantum era," *National Science Review*, vol. 8, no. 9, 07 2021, nwab115.

[70] F. Guo, W. Susilo, Y. Mu, F. Guo, and W. Susilo, "Notions, definitions, and models," *Introduction to Security Reduction*, pp. 5–12, 2018.

[71] S. Mella and R. Susella, "On the homomorphic computation of symmetric cryptographic primitives," in *Proceedings of the 14th IMA International Conference on Cryptography and Coding - Volume 8308*, ser. IMACC 2013. Berlin, Heidelberg: Springer-Verlag, 2013, p. 28–44.

[72] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "Tfhe: fast fully homomorphic encryption over the torus," *Journal of Cryptology*, vol. 33, no. 1, pp. 34–91, 2020.

[73] A. A. Bendoukha, O. Stan, R. Sirdey, N. Quero, and L. Freitas, "Practical homomorphic evaluation of block-cipher-based hash functions with applications," in *International Symposium on Foundations and Practice of Security*. Springer, 2022, pp. 88–103.

[74] B. Wei, R. Wang, Z. Li, Q. Liu, and X. Lu, "Fregata: Faster homomorphic evaluation of aes via tfhe," in *International Conference on Information Security*. Springer, 2023, pp. 392–412.

[75] S. Hirose, "Some plausible constructions of double-block-length hash functions," in *Fast Software Encryption*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 210–225.

[76] S. Halevi and V. Shoup, "Design and implementation of HElib: a homomorphic encryption library," *Cryptology Archive*, 2020.

[77] G. Hofemeier and R. Chesebrough, "Introduction to intel aes-ni and intel secure key instructions," *Intel, White Paper*, vol. 62, 2012.

[78] T. Pöppelmann, T. Oder, and T. Güneysu, "High-performance ideal lattice-based cryptography on 8-bit atxmega microcontrollers," in *International conference on cryptology and information security in Latin America*. Springer, 2015, pp. 346–365.

[79] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *International conference on the theory and application of cryptology and information security*. Springer, 2001, pp. 514–532.

[80] K. Piotrowski, P. Langendoerfer, and S. Peter, "How public key cryptography influences wireless sensor node lifetime," in *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, 2006, pp. 169–176.

[81] J. Howe and B. Westerbaan, "Benchmarking and Analysing the NIST PQC Finalist Lattice-Based Signature Schemes on the ARM Cortex M7, Paper 2022/405," *Cryptology ePrint Archive*, 2022.

[82] O. Papadakis, M. Papadimitriou, A. Stratikopoulos, M. Xekalaki, J. Fumero, N. Foutris, and C. Kotselidis, "Towards gpu accelerated fhe computations," in *2024 IEEE International Conference on Cyber Security and Resilience (CSR)*. IEEE, 2024, pp. 694–699.

[83] D. Pei, A. Salomaa, and C. Ding, *Chinese remainder theorem: applications in computing, coding, cryptography*. World Scientific, 1996.

ATTILA ALTAY YAVUZ is an Associate Professor at the Bellini College of Artificial Intelligence, Cybersecurity, and Computing at the University of South Florida (USF), where he also directs the Applied Cryptography Research Laboratory. Previously, he was an Assistant Professor at Oregon State University (2014–2018) and USF (2018–2021), following his role as a research scientist at the Robert Bosch Research and Technology Center North America (2011–2014). He holds a Ph.D. in Computer Science from North Carolina State University (2011) and an M.S. from Bogazici University (2006). Dr. Yavuz's broad research interests center on designing, analyzing, and deploying cryptographic techniques to strengthen the security of computer systems and next-generation networks. His work has been recognized with numerous honors, including the NSF CAREER Award, multiple research awards from Bosch (five) and Cisco (four), three USF Excellence in Research Awards, several major federal grants, and numerous best paper awards. His research leadership extends to editorial board service (e.g., IEEE TDSC) and organizing roles in major conferences (e.g., ACM CCS). His work encompasses 115 peer-reviewed publications in top-tier venues (e.g., Usenix, NDSS, CCS, IEEE TIFS), patents, and technology transfers to industry partners, particularly in searchable encryption and intra-vehicular network security, impacting tens of millions of users worldwide. He is a Senior Member of the IEEE, the National Academy of Inventors, and ACM.

SALEH DARZI is a Ph.D. Candidate in Computer Science and Engineering Department, actively engaged in research within the Applied Cryptography Research Laboratory (ACRL) under the supervision of Dr. Attila Yavuz at the University of South Florida. His primary research pursuits revolve around post-quantum and applied cryptography, with a focus on addressing challenges in the privacy and security of IoT, Blockchain technology, and network security. Saleh holds a Master of Science degree in Electrical Engineering (Communication-System) from K. N. Toosi University of Technology, Tehran, Iran, obtained in 2021.

SAIF EDDINE NOUMA is a Ph.D. candidate at the Bellini College of Artificial Intelligence, Cybersecurity, and Computing, at the University of South Florida. He earned his bachelor of Engineering from École Polytechnique de Tunisie, Tunisia, in 2020. His research focuses on lightweight and post-quantum cryptography tailored for the Internet of Things (IoT) and digital twins.