Lightweight and Scalable Post-Quantum Authentication for Medical Internet of Things

Attila A. Yavuz, Saleh Darzi and Saif E. Nouma

University of South Florida, Department of Computer Science and Engineering, Florida, USA.

ARTICLE INFO

Keywords:
Medical Internet of Things
Digital Signatures
Lightweight Authentication
Post-Quantum Security
E-Health

ABSTRACT

The Medical Internet of Things (MIoT) harbors resource-limited medical embedded devices that collect security-sensitive data from users for analysis, monitoring, and diagnosis, often involving cloud services. Digital signatures play a foundational role in ensuring the authentication and integrity of this sensitive medical information, critical for the trustworthiness of large-scale MIoT applications. However, traditional signatures used in current IoT systems may lack the necessary long-term security and are vulnerable to emerging quantum computer threats. NIST's post-quantum cryptography (PQC) standards, though promising, impose heavy overhead unsuitable for battery-limited MIoT devices. Efforts to design more computationally efficient post-quantum (PQ) signatures have faced challenges, either introducing significant memory overhead and potential vulnerabilities (e.g., side-channel) or relying on strong assumptions (e.g., central trusted servers or semi-honest non-colluding servers), which may not align well with highly regulated healthcare applications. Hence, there is a need for highly lightweight PQ secure signatures that prioritize the strict resource limitations of embedded MIoT devices without imposing strong security assumptions or extra architectural requirements. This paper introduces `INFinity-HORS` (`INF-HORS`), a lightweight PQ digital signature. To the best of our knowledge, `INF-HORS` is the first signer-optimal hash-based signature offering polynomial unbounded signing capabilities under minimal architectural assumptions. Unlike other PQ signatures, `INF-HORS` does not require hyper-tree structures or incur the high memory usage seen in multivariate counterparts. Our performance analysis confirms that `INF-HORS` is significantly more computationally efficient than NIST PQC standards like *Dilithium* and *SPHINCS+*, while maintaining a compact memory and signature footprint. We prove `INF-HORS`'s security in the random oracle model and show through experiments that it achieves 20× faster signature generation and smaller signature and private key sizes compared to BLISS-I on an 8-bit ATxmega128A1 microcontroller. `INF-HORS` does not rely on non-colluding verification servers, secure enclaves, or trusted verification assisting entities, minimizing security risks and making it ideal for extending battery life in MIoT with minimal cryptographic overhead and strong security assumptions.

1. Introduction

Recent advances in next-generation networked systems [47] and digital twin technologies [71] have paved the way for the proliferation of the Internet of Things (IoT), which encompasses billions of sensing and computing devices. IoT is increasingly receiving significant attention from both industry and academia due to its capability to realize autonomous systems and virtual reality applications with minimal human intervention. Numerous domains, namely healthcare [3], industry [47], and military [56], have benefited from IoT support. For example, smart-health applications [3] depend on a wide array of resource-limited devices (e.g., wearable devices, medical implants [37, 1]). These devices collect security-sensitive information about patients and the infrastructure to facilitate critical decision-making processes. Healthcare companies manufacture cardiac pacemakers to be implanted in patients [1]. They actively transmit sensitive heartbeat data for monitoring [78] and to prevent heart attacks [69]. Such IoT systems require lightweight cybersecurity mechanisms capable of supporting a massive number of devices while respecting the resource limitations (i.e., memory, processing, and bandwidth) and scalability requirements of next-generation networks [9].

In medical IoT applications, authentication and integrity are essential cybersecurity services that safeguard sensitive data against critical attacks such as man-in-the-middle attacks and data tampering [46]. Such attacks warrant serious consideration, particularly when it comes to safeguarding highly sensitive information, such as health-related data (e.g., heartbeat rate) or security information (e.g., audit logs). Insecure data authentication directly undermines the integrity of information collected from implantable devices. Therefore, these devices not only lose their utility but also pose potential deadly damage to the patient (e.g., unattended slow heartbeat correction) [60]. Digital signatures offer public verifiability, non-repudiation, and scalability and are essential tools to achieve authentication and integrity in IoT systems [49]. The non-repudiation prevents an authenticator from denying its signature. Therefore, it is important for potential legal disputes (e.g., [17, 59]). Public verifiability permits external audits (both for medical data and device logs [16, 77, 29]) that are important requirements for medical IoTs. Moreover, digital signatures offer scalability via public key infrastructures and are useful for IoT applications on a large number of devices. However, despite their merits, the use of digital signatures on resource-limited (embedded)

attilaayavuz@usf.edu (A.A. Yavuz); salehdarzi@usf.edu (S. Darzi); saifeddinenouma@usf.edu (S.E. Nouma)
ORCID(s):

[1] https://www.neuromodulation.abbott/us/en/healthcare-professionals/hcp-chronic-pain.html

medical IoT applications has various challenges and requirements. Below, we list some of the most desirable properties that a digital signature scheme must achieve to be suitable for deployment on embedded medical IoT devices:

• *Post-Quantum Cryptography (PQC) for Long-term Se--curity*: The rapid emergence of quantum computing intensifies security challenges as traditional digital signatures based on conventional number theoretic assumptions (e.g., factorization problem, discrete logarithm problem) are vulnerable to quantum attacks [65, 6]. This vulnerability has garnered considerable attention from industry, academia, and government sectors [22, 19, 2]. Achieving quantum-safe security for IoT is paramount yet challenging due to the computationally costly operations in PQ cryptographic algorithms [51] and resource limitations of low-end IoT devices (e.g., an 8-bit microcontroller).

• *Computational Efficient Signature Generation for Min--imal Energy Usage*: Many of the embedded medical IoT devices are battery powered (e.g., [17]). It is critical for these devices to extend their battery life-span since, for example, in wearables, this translates into higher usability [16], while in some implantable devices, it directly impacts the patient's life quality (e.g., a replacement may require surgical intervention [17]). Hence, it is highly desirable that the underlying cryptographic mechanism consumes low energy, thereby making a minimum impact on the battery life. Yet, it has been shown that some of the standard digital signatures (e.g., [4, 42]), even only with conventional security (e.g., based on Elliptic Curve Cryptography (ECC)), can negatively impact the battery life of low-end IoT devices [50, 51]. The signature energy consumption issue compounds when PQ security is considered [64, 19] (as discussed in 1.1).

• *Minimal Cryptographic Memory and Bandwidth Usage*: *(i)* Embedded IoT devices often have a compact memory space. For example, an 8-bit Micro-Controller Unit (MCU) such as the widely used ATxmega128A1 comes equipped with only 128 KB of static flash memory [2]. Given these constraints, an ideal lightweight signature should offer highly compact key sizes as well as minimum memory expansion during the signature computation. *(ii)* The cryptographic memory usage is also impacted by the code size required to execute the digital signature. A simple code base can be achieved by running basic cryptographic operations (e.g., hashing) without incurring complex operations (e.g., EC scalar multiplication [21], sampling [25]). This approach not only helps to manage the limited computational resources and memory capacity of IoT devices, but also offers ease of implementation and lowers energy usage with minimal memory access. *(iii)* The size of the signature is another important criterion since it not only affects memory usage, but also bandwidth overhead with an impact on energy consumption (e.g., the larger the wireless cryptographic transmission, the higher the battery consumption [62]). Note that PQ secure digital signatures have notoriously large key and signature sizes (see Section 1.1), making them even more difficult to deploy on low-end IoT devices.

• *Minimal Security Assumptions for Improved Security and Robustness*: *(i)* Lightweight cryptographic primitives may be attained if one accepts additional security assumptions in exchange for a better performance. For example, assuming semi-honest and non-colluding servers or the presence of secure enclaves have been shown to substantially increase the signature generation performance for low-end devices [51, 9]. Yet, considering the highly security sensitive nature of medical applications, it is desirable to avoid such additional assumptions, thereby offering long-term trust not only with PQC but also by reducing extra architectural assumptions. *(ii)* PQC digital signatures involve complex operations such as Gaussian sampling [26], rejection sampling [25], and complex arithmetics [24, 64] that can increase the susceptibility of digital signatures to side-channel attacks [38, 67]. These risks are exacerbated on embedded architectures due to the increased difficulty of countermeasures (e.g., [16]), and low-end IoT devices are also known to be susceptible to attacks due to low-quality random number generations that itself led to various potential attacks [55]. It is highly desirable to develop digital signatures that can avoid all these hurdles to supplement the long-term trust premise sorely needed by medical IoTs.

Our literature review reveals a significant gap in simultaneously achieving all the desirable properties for digital signatures in low-end (embedded) IoT environments. In particular, the substantial signing overhead with current post-quantum signatures poses a severe limitation on their practical deployment in IoT devices. In the following, we first outline the research gap in the state-of-the-art. We then summarize our contributions to address these gaps.

1.1. Related Work and Limitations of the State-of-the-Art

In this section, we discuss the state-of-the-art digital signatures relevant to our context, focusing on those that offer computationally efficient signing and feature small private key and signature sizes. Additionally, we target signatures that provide post-quantum (PQ) security. Given the plethora of digital signature schemes proposed in the literature, we first briefly review prominent conventional signatures. Subsequently, we shift our focus to post-quantum signatures, including both standardized schemes and those noted for their signing efficiency.

Conventional lightweight signatures: They can offer efficient signature generation and small key sizes, along with additional security guarantees (e.g., [73, 50, 68, 18]). The signature schemes based on the seminal elliptic-curve (EC) Schnorr [21] are notable for their efficiency compared to other conventional signature categories, such as pairing-based [41] and factorization-based [76]. For example, a recent EC-based signature scheme [50] is signer-efficient and provides single-signer signature aggregation. In a different approach, Chen et al. [18] combine confidentiality with authentication. However, note that despite their merits, none

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

of these schemes or other relevant ECC-based conventional signatures provide PQ security.

Standard PQ signatures: NIST recently standardized Dilithium as the lattice-based digital signature standard (FIPS 204) and SPHINCS+ as the stateless hash-based digital signature standard (FIPS 205) [48].

Hash-based digital signatures rely on minimal intractability assumptions and therefore offer strong security guarantees. The stateless hash-based SPHINCS+ [12], is based on a variant of the one-time signature scheme HORS [58] and utilizes a hyper-tree structure to construct a multiple-time signature. Although SPHINCS+ achieves PQ security with stronger security assumptions, it has a signing time and a signature that are slower and smaller than those of ECDSA by order of magnitudes, respectively. Consequently, it is not suitable for resource-constrained IoT devices. Consequently, it is not suitable for resource-constrained IoT devices. Stateful hash-based signatures (e.g., RFC standard XMSS-MT [34], RFC standard LMS [43]) offer similar security guarantees since they rely on a HORS variant (e.g., W-OTS [45]). They also come with advanced security properties (e.g., forward security). However, they require state management and are not practical for resource-limited devices due to the high computational cost and high memory usage on the signer side.

Lattice-based signatures rely on module lattice problems (e.g., learning with errors LWE [25]) and offer a better efficiency balance between signing and verification algorithms. For instance, the two lattice-based schemes selected by NIST, Dilithium [25] and Falcon [26], offer smaller signature sizes and faster signing than their hash-based counterpart, SPHINCS+. However, despite their efficiency, these lattice-based signature schemes are not ideally suited for resource-constrained IoT devices either. They require more complex computations and have larger signature sizes compared to conventional signatures. Furthermore, lattice-based signatures involve techniques such as Gaussian and/or rejection sampling, which are vulnerable to side-channel attacks (e.g., [38]). To date, there are no deployable open-source implementations of lattice-based digital signatures that are suitable for highly resource-limited embedded devices, such as 8-bit microcontrollers, with the exception of BLISS [24], which was not selected as a NIST PQC standard [74] and proved to be prone to side-channel attacks [67].

Additional PQ signatures for standardization: NIST also launched an additional competition, alongside standardized schemes, to encourage diversity in signature standardization, emphasizing fast verification and short signatures. The submissions are heterogeneous and cover all PQC categories, including code-based signatures (e.g., [7]), mutlivariate-based (e.g., [13]), and symmetric-based ([39]) signatures. In particular, there exist numerous lightweight multivariate-based digital signatures (e.g., [40, 63, 64]) that offer computationally efficient signature generation and a small signature size. For example, Shim et al. [64] provide advantages over the NIST PQC standards with an efficient signing and small signature size, but it still entails large private key and code sizes compared to conventional counterparts. Its private key is 12.6 KB which is an order of magnitude larger than ECDSA while its code size occupies 62.6% of the total flash memory size for an ATxmega128A1 MCU. Therefore, it incurs costly memory usage on low-end IoT devices (e.g., 8-bit MCU). Despite the PQC progress and the real-world applications, addressing the computational demands and memory usage of these signature schemes remains a significant challenge for meeting the requirements of low-end IoT devices, often operate on 8-bit MCUs [74]. In addition to consuming a significant amount of memory and stack resources, several multivariate signatures are vulnerable to various attack vectors aimed at breaking the unforgeability of proposed signatures within polynomial time [66, 30].

Lightweight PQ signatures with additional assumptions: They offer highly efficient signature generation but require additional assumptions (e.g., [9, 51, 53]). For example, ANT [9] is a lattice-based digital signature that delegates costly generated commitments to a set of distributed, non-colluding, and semi-honest servers [61]. Additionally, verifiers must communicate with these servers before verification, which may lead to network delays and outage risks. Another approach involves Trusted Execution Environment (TEE)-assisted signatures, which delegate the burden of costly computations to a TEE-enabled server (e.g., [53, 49, 51]). For example, HASES [49] and its extension [51] are hash-based signatures derived from the one-time HORS [58], utilize a single TEE-enabled cloud server to provide one-time public keys to verifier. However, reliance on single TEE-enabled servers introduces a centralized root of trust and potential key escrow problems. Given these additional assumptions, such digital signatures might not be ideal for certain IoT applications, as they deviate from traditional public key settings.

There is a pressing demand for efficient post-quantum digital signatures that are tailored to meet the performance and security constraints of IoT applications. An ideal digital signature for IoT must meet the above desirable properties without relying on assumptions that may not hold in diverse real-world scenarios, all while maintaining a lightweight design. In this work, we aim to answer the following research questions:

(i) Is it possible to achieve an efficient post-quantum signature generation that meets the stringent requirements of IoTs in terms of memory, processing, and bandwidth? (ii) Is it possible to permit an efficient signing without imposing non-conventional and risky assumptions on the verifier, like non-colluding multiple servers or third parties storing secret keys, suffering from key escrow and single root of trust? (iii) Can we meet these requirements in a multi-user setting to scale IoTs?

1.2. Our Contribution

We created a novel lightweight post-quantum signature named `INFinity-HORS` (`INF-HORS`). Our key observation is that the main overhead of hash-based signatures stems from the management of one-time public keys. To address this problem, we develop innovative strategies that permit verifiers to construct one-time public keys without interacting with the signers or any other third parties. INF-HORS transforms one-time HORS into a multiple-time signature by permitting verifiers to extract one-time keys from a master public key via encrypted pseudo-random function evaluations. To the best of our knowledge, this is the first HORS-type (hash-based) scheme that lifts the limitation of number of messages to be signed without having special assumptions like semi-honest servers (ANT [9]), trusted hardware (HASES [49]) or costly signing and large key sizes (XMSS [35], SPHINCS+ [12]). Our design offers several desirable properties, as follows:

• ***Signer Computational/Energy Efficiency***: Our scheme makes only one call to the one-time signature scheme, HORS, with identical signature sizes, and it is *optimal* with respect to HORS. Hence, INF-HORS performs only a small-constant number (e.g., 16) of Pseudo-Random Function (PRF) calls per signing. Table 3 shows that INF-HORS signature generation is 19.5 times and 1130 times faster than NIST PQC standards Dilithium-II and Falcon-512, respectively. It is even significantly more efficient than the best conventional-secure (ECC-based) signatures. As shown in Section 6, this efficiency directly translates into substantial energy savings.

• ***High Memory and Bandwidth Efficiency***: The private key of INF-HORS consists solely of a single random seed (e.g., 128-bit), and it transmits only one HORS signature per message (i.e., 256 bytes). Thus, INF-HORS has the smallest private key and the most compact signature among its PQ counterparts (see Table 3). Unlike previous works, such as multivariate-based [64], lattice-based [24]) the code size of our signature generation is also minimal. INF-HORS signing involves only few PRF calls and one hash call and avoids expensive operations (e.g., EC scalar multiplication [68], sampling operations [67]). Moreover, our selection of AES-128 as the PRF and SHA-256 as the cryptographic hash function aligns with symmetric standards, promoting standard compliance and ease of implementation [51]. This alignment facilitates the transition to PQC.

• ***Advanced Security Features***: *(i)* INF-HORS follows the standard public key setting. Unlike some PQ alternatives [9], it refrains from unconventional assumptions like non-colluding servers. Similarly, INF-HORS avoids security assumptions like trusted private key servers to ferry public keys, which introduce architectural risk. *(ii)* The Gaussian and rejection sampling operations in lattice-based signature schemes are highly prone to side-channel and timing attacks (e.g., [38]). INF-HORS only relies on symmetric cryptographic primitives and is therefore free from these types of attacks. Moreover, the signature generation of INF-HORS do not generate random keys and therefore can mitigate the vulnerabilities stemming from weak random number generators typically found in resource-constrained IoTs.

• ***Compact Multi-User Storage***: INF-HORS allows a verifier to construct a target one-time public key for any given valid signer identity and state, from a constant-size master public key. This approach allows for compact storage for a large number of users without the need to maintain individual public keys and certificates for each user (e.g., 2^{20} users).

• ***Online/Offline Verification***: INF-HORS enables verifiers to derive one-time public keys from the master public key either on-demand or prior to verification. The public key construction is computationally costly due to the encrypted function evaluations. However, it can be done before receiving signatures. It can be run by the verifier stand-alone or offloaded to a resourceful cloud server. Hence, the overhead of public key construction is substantially mitigated in practice.

All these properties indicate that INF-HORS is an ideal post-quantum signature to achieve lightweight and scalable authentication for resource-limited IoT applications.

Limitations and Discussion on Potential Applications: The signature verification of INF-HORS is computationally costlier than the NIST PQC digital signatures with a larger public key. Therefore, INF-HORS is not suitable for delay-aware applications that demand immediate verification (e.g., real-time authentication in vehicular [8] or smart-grid systems) or for resource-limited verifiers.

At the same time, we have demonstrated that INF-HORS offers an ideal performance portfolio for IoT applications requiring a non-interactive signer that prioritizes near-optimal signing, small code footprint, minimal private key and signature sizes, and ease of implementation. Consequently, we anticipate that INF-HORS is well-suited for applications where near-optimal signer performance is critical and some delay and storage in signature verification can be tolerated. We consider that a cloud-supported embedded medical IoT device application (e.g., with medical wearables and/or implants), as elaborated further in Section 3 (see Figure 1), is a proper representative of a heterogeneous IoT use-case (e.g., [72]) that INF-HORS can serve the best. In these applications, the efficiency and battery longevity of the embedded medical device are the utmost priorities, wherein available computation, memory, and bandwidth of the already resource-limited devices must be ideally dedicated to the healthcare application and to expensive PQC cryptographic operations. Moreover, long-term security with PQC, minimization of relevant PQC side-channel vectors, and elimination of semi-honest verifiers and secure enclaves are also important features for such high-security medical IoTs. Finally, the telemetry and signatures are usually collected by a resourceful cloud that has all the necessary computation and storage resources to verify INF-HORS signatures. Overall, we posit that INF-HORS can be an ideal digital signature for such heterogeneous IoT applications that prioritize high-security, PQC, and optimal signer efficiency with delay-tolerant and resourceful verifiers.

2. Preliminaries

The notations and acronyms are described in Table 1.

Definition 1 Hash to Obtain Random Subset (HORS) [58] is a one-time digital signature comprised of three algorithms:

Table 1
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
sk/PK	Private/Public key
msk/MPK	Master private/public key
ID_i/N	User identity (e.g., MAC address) and 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
$x \overset{\$}{\leftarrow} \mathcal{S}/\lvert x\rvert$	random selection from a set $\mathcal{S}$ and bit length of variable x
$\vert\vert/\oplus$	string concatenation and 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

- $(sk, PK, I_{\texttt{HORS}}) \leftarrow \texttt{HORS.Kg}(1^\kappa)$: 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 $v_i \leftarrow f(s_i), \forall i = 1, \dots, t$. Finally, 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 $v_{i_j} = f(\sigma_j), \forall j = 1, \dots, k$, it returns $b = 1$, otherwise $b = 0$.

Definition 2 A Fully Homomorphic Encryption scheme (FHE) [5] 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.
- $C \leftarrow \texttt{FHE.Eval}(PK', \mathcal{F}(\vec{c} = \{c_j\}_{j=1}^n))$: Given PK', a function $\mathcal{F}$, and a set of input arguments $\vec{c}$, it evaluates $\mathcal{F}$ on $\vec{c}$ 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 [15]. Note that these FHE instantiations also have a post-quantum security premise [75].

The Davies-Meyer scheme ($\texttt{DM}$) [57] is an iterated cryptographic hash function based on a block cipher. In $\texttt{INF-HORS}$, we only rely on the one-wayness ($\texttt{OWF}$) of $\texttt{DM}$, which is based on the IND-CPA security of the symmetric cipher E.

Definition 3 $B_n \leftarrow \texttt{DM}(M, B_0)$: Given a message $M = \{m_i\}_{i=1}^n$ with n blocks, a pre-defined initial value $I_{\texttt{DM}} = B_0$, block cipher E of length k, it computes $n = \lceil \frac{|M|}{k} \rceil$, and $B_i = E_{m_i}(B_{i-1}) \oplus m_i, \forall i = 1, 2, ..., n$. It outputs B_n.

Definition 4 A Public Key Outsourced Signature scheme $\texttt{PKO-SGN} = (\texttt{Kg}, \texttt{Sig}, \texttt{PKConstr}, \texttt{Ver})$ is 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 $sk = \vec{\gamma}$, and the system-wide parameters $I \leftarrow I_{\texttt{FHE}}$.
- $\sigma_i^j \leftarrow \texttt{PKO-SGN.Sig}(\gamma_i, M_j)$: Given the seed $\gamma_i \in \vec{\gamma}$ 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.

3. Models

System Model: We follow the traditional public-key-based broadcast authentication model that is designed for delay-tolerant wearable and heterogeneous IoT applications. Figure 1 depicts the entities of our system model, described as follows:

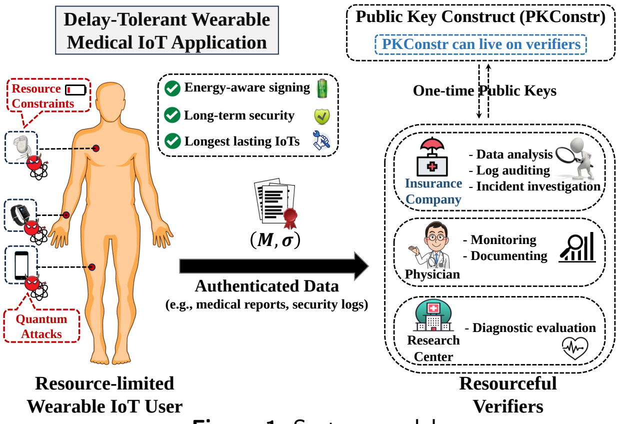

Figure 1: System model

- *Signer:* is a resource-constrained IoT device such as a smart watch, medical pacemaker, or implantable medical device [17]. In this context, we focus on a wearable medical IoT application. However, our proposed digital signature is also suitable for other potential use cases, such as industrial or military applications. The role of the signer is limited to signing generated messages and broadcasting them to verifiers. These messages may entail data specific to the wearable IoT device, such as heart rate information from a medical pacemaker or security logs for future audit and/or analysis at the verifier. Our digital signature prioritizes efficient and energy-aware computation on the signer over the efficiency of signature verification. This emphasis is due to the resource limitations of the device, such as battery life, processing power, and memory capacity.

• *Verifier:* is a resourceful device (e.g., physician, authority). He is the recipient of to-be authenticated messages from signers. The verifier is capable of constructing one-time public keys from the master public key (MPK). From MPK, the verifier is able to derive any public key PK_i^j for any user with identity ID_i in a network of (e.g., $N = 2^{20}$) signers. Unlike existing models that rely on additional architectural entities (e.g., semi-honest non-colluding servers [9] or a trusted parties [52, 53]), our scheme does not require a third party and can perform verification by itself. In section 6, we discuss optional alternatives where verifiers can outsource public key construction to a resourceful entity (e.g., a cloud server). The public key derivation involve FHE computations which is generally expensive and is not real-time. Hence, our digital signature is limited to delay-tolerant applications where verification is not real-time.

Threat and Security Model: Our threat model is based on an 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 [27]. It captures a Probabilistic Polynomial Time (PPT) adversary ($\mathcal{A}$) aiming at forging message-signature pairs. $\mathcal{A}$ is able to run passive and active attacks. The EU-CMA experiment is defined as follows:

Definition 5 The EU-CMA experiment $Expt_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}$ for an PKO-SGN digital signature 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 $\texttt{PKO-SGN.Ver}(PK, M^*, \sigma^*) \ == \ 1$ 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 $Adv_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}(t, q_s) = \max_{\mathcal{A}}\{Adv_{\texttt{PKO-SGN}}^{\texttt{EU-CMA}}(\mathcal{A})\}$, 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 PKO-SGN.PKConstr(.) are as follows:

1. *Signing oracle* $\texttt{PKO-SGN.Sig}_{sk}(.)$: Given an input message M, it output a signature $\sigma \leftarrow \texttt{PKO-SGN.Sig}_{sk}(M)$.
2. Public key construct oracle 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., [49, 9]), PKO-SGN.PKConstr(.) does not require a root of trust on introduced entities (e.g., [9, 51]) and can be run based on public key data. PKO-SGN.PKConstr(.) may be run by the verifier or a resourceful third party.

4. The Proposed Scheme

We first present our proposed scheme, INF-HORS. We then describe its instantiations, design rationale, and optimizations. The main bottleneck of hash-based digital signatures is the generation and management of one-time public keys. As outlined in Section 1.1, 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., [53]). 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.

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 without the risk of private key compromises. Our approach effectively transforms one-time HORS into practically unbounded hash-based signature, and therefore, fittingly, we name our new scheme INFinity-HORS (INF-HORS). We provide the details of INF-HORS in Algorithm 1.

The key generation algorithm INF-HORS.Kg, first derives the master signing key msk and sets up the public parameters I including HORS, FHE, and DM parameters as in Definition 1, 2, 3, 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 INF-HORS public key as $PK = (PK', MPK)$ (Step 5). 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 6).

The signature generation INF-HORS.Sig, for the signer ID_i, first derives the private key sk_i^j from the seed γ_i for a given message state (counter) j (Step 1). The rest of signing is as in HORS.Sig, with the difference that we compute the signature elements $\{s_i^{j,\ell}\}_{\ell=1}^{k}$ via PRF evaluations based on sk_i^j, instead of random generations (Step 2-4). Finally, the

Algorithm 1 INFinity HORS (INF-HORS) Scheme

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

$\sigma_i^j \leftarrow$ INF-HORS.Sig($\gamma_i, M_i^j$): The signer $ID_i$ computes a signa-
ture on a message $M_i^j$ as follows:
1: $sk_i^j \leftarrow$ 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 $\ell = 1, \dots, k$ do
4:     $s_i^{j,\ell} \leftarrow$ PRF($sk_i^j, x_i^{j,\ell}$)
5: Set $j \leftarrow j + 1$
6: return $\sigma_i^j = (s_i^{j,1}, s_i^{j,2}, ..., s_i^{j,k}, j)$

$cv_i^j \leftarrow$ INF-HORS.PKConstr($PK, ID_i, j$): Performed by the veri-
fier 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: $c\gamma_i \leftarrow$ FHE.Eval($PK'$, PRF($MPK, ID_i$))
2: $csk_i^j \leftarrow$ FHE.Eval($PK'$, PRF($c\gamma_i, j$))
3: for $\ell = 1, \dots, t$ do
4:     $cv_i^{j,\ell} \leftarrow$ FHE.Eval($PK', f$(PRF($csk_i^j, \ell$)))
5: return $cv_i^j \leftarrow (cv_i^{j,1}, cv_i^{j,2}, ..., cv_i^{j,t})$

$b_i^j \leftarrow$ INF-HORS.Ver($PK, M_i^j, \sigma_i^j$):
1: $cv_i^j \leftarrow$ INF-HORS.PKConstr($PK, ID_i, j$)
2: Execute Step 2 in INF-HORS.Sig
3: for $\ell = 1, \dots, k$ do
4:     $v_i^{j,\ell} \leftarrow f(s_i^{j,\ell})$
5:     $CV_\ell^j \leftarrow$ FHE.Enc($PK', v_\ell^j$)
6:     $b_i^{j,\ell} \leftarrow$ FHE.Eval($PK'$, CMP($cv_i^{j,x_i^{j,\ell}}, CV_i^{j,\ell}$))
7: if $b_i^{j,\ell} = 1, \forall \ell = 1, \dots, k$ then, return $b_i^j = 1$ else, return
   $b_i^j = 0$
```

signer updates the state j and discloses the HORS signature (Step 5).

INF-HORS.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 used 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 (Step 4-5).

The signature verification INF-HORS.Ver resembles HORS.Ver, but starts by constructing public keys using INF-HORS.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 4.1, the verifier may construct public keys offline before receiving the message-signature pair. Additionally, to reduce the storage demands, the verifier may use an alternative method by providing the indices (i.e., $\{x_i^{j,\ell}\}_{\ell=1}^k$ in Step 2, INF-HORS.Sig) instead of the counter j to the INF-HORS.PKConstr routine.

4.1. INF-HORS Instantiations and Optimizations

The generic INF-HORS 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 [23]. We instantiated our FHE with BGV cryptosystem [15] 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. (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 INF-HORS.PKConstr requires FHE evaluations, we require FHE-friendly cryptographic primitives that suit the needs of INF-HORS. 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 [44]. Recently, several research efforts have concentrated on the homomorphic evaluation of hash functions such as SHA256, SM3, etc., utilizing FHE schemes like TFHE [20] that enable rapid bootstrapping. However, they remain considerably distant from practical application, with execution times on the order of minutes [10, 70].

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 cryptographic 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) [57], 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 [31]), 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 decided that AES is a suitable choice for our DM instantiation: (i) It is widely deployed with several optimized implementations. (ii) It has a low number of rounds with no integer operations. (iii) The AES circuit has an algebraic structure that is compliant with parallelism, packing techniques, and GPU optimizations [44]. (iv) Compared to other hash functions, the AES-based DM has a smaller and fixed-size memory to store hash values iteratively. (v) Finally, homomorphic evaluation of AES has been well-studied and available in existing libraries (e.g., HElib [28]).

Optimizations: We introduce online-offline optimizations to permit an efficient signature verification. (i) The public key construction is independent of messages to be verified and can be executed for any ID_i and state information beforehand. Therefore, the verifier can run INF-HORS.PKConstr with batch processes offline, and use these encrypted public keys to efficiently verify signatures online. As shown in Section 6, this offers tremendous performance gains for the online verification. (ii) Instead of generating full-set of t keys, the verifier can only construct k one-time public key components required for verification, thereby reducing the numbers of FHE evaluations. (iii) Recall that INF-HORS.PKConstr does not take any private input and can be publicly executed by any entity. Therefore, optionally, the verifier can offload offline execution of INF-HORS.PKConstr to a resourceful entity (e.g., cloud server). In exchange for a transmission delay, this approach can lift the major burden of FHE evaluations from the verifier, while enabling the resourceful entity to employ several parallelization and GPU-acceleration capabilities that are amenable to our INF-HORS instantiations.

5. Security Analysis

We prove that INF-HORS is EU-CMA secure as follows.

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

Proof: Let $\mathcal{A}$ be the INF-HORS attacker. We construct a simulator $\mathcal{F}$ that uses $\mathcal{A}$ as a subroutine to break one-time EU-CMA secure HORS, where $(\overline{sk}, \overline{PK}, I_{\texttt{HORS}}) \leftarrow \texttt{HORS.Kg}(1^{\kappa})$ (Definition 1). $\mathcal{F}$ 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 $\texttt{INF-HORS.Sig}_{sk}(.)$ outputs. $\mathcal{F}$ randomly chooses a target forgery index[*] $w \in [1, q_s]$. $\mathcal{A}$ uses a user identity $ID_i \in \overrightarrow{ID}$, where $i \overset{\$}{\leftarrow} \{1, \ldots, N\}$.

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

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

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

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

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

(1) $\texttt{INF-HORS.Sig}_{sk}(.)$*:* $\mathcal{F}$ returns $\sigma_i^w \leftarrow \texttt{HORS.Sig}_{\overline{sk}}(M_i^w)$ by querying HORS signing oracle, if $j = w$. Otherwise, $\mathcal{F}$ runs the steps (2-5) in INF-HORS.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) INF-HORS.PKConstr(.) *Queries:* If $j = w$ then $\mathcal{F}$ returns $cv_i^w = \texttt{FHE.Enc}(PK', \overline{PK})$. Otherwise, $\mathcal{F}$ returns cv_i^j.

- *Forgery of $\mathcal{A}$:* $\mathcal{A}$ produces a forgery (M^*, σ^*) on PK. $\mathcal{A}$ wins the EU-CMA experiment if $\texttt{INF-HORS.Ver}(PK, M^*, \sigma^*) == 1$ 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 INF-HORS, $\mathcal{F}$ also fails to win the EU-CMA experiment for HORS. As a result, $\mathcal{F}$ *aborts* and returns 0. Otherwise, $\mathcal{F}$ checks if $\texttt{HORS.Ver}(\overline{PK}, M^*, \sigma^*) == 1$ and M^* was not queried to the HORS signing oracle (i.e., $\texttt{HORS.Sig}_{\overline{sk}}(.)$). 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 $PK_i^w = \texttt{FHE.Enc}(PK', \overline{PK})$ 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

[*] We follow SPHNICS+ [12] where the maximum number of signing queries is $2^{40} \leq q_s \leq 2^{60} \ll 2^{\kappa}$

during the simulation, which occurs with a negligible probability in terms of κ due to the correctness property of $\texttt{FHE}$. Therefore, we conclude $Pr[\overline{Abort1}] \approx 1$.

(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 $\texttt{INF-HORS}$ is $Pr[Forge|\overline{Abort1}] = Adv^{\texttt{EU-CMA}}_{\texttt{INF-HORS}}(q_s, t)$. 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 [58]. (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 [15]. (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 $\texttt{DM}$ by breaking the underlying IND-CPA cipher, which also happens with negligible probability in terms of κ [57]. 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 $\texttt{EU-CMA}$ experiment with $Adv^{\texttt{EU-CMA}}_{\texttt{HORS}}(t', q_s')$: $Pr[Win] = Pr[\overline{Abort1}] \cdot Pr[Forge|\overline{Abort1}] \cdot Pr[\overline{Abort2} \ |\overline{Abort1} \wedge Forge]$. Therefore, $Pr[Win]$ is bounded as:

$$Adv^{\texttt{EU-CMA}}_{\texttt{INF-HORS}}(t, q_s) \quad \leq \quad q_s \cdot Adv^{\texttt{EU-CMA}}_{\texttt{HORS}}(t', q_s')$$

- *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 $k \cdot \texttt{PRF}(.)$; and each $\texttt{INF-HORS.PKConstr}(.)$ query needs $(t+2) \cdot \texttt{FHE.Eval(PRF)}$. The approximate running time of $\mathcal{F}$ is $\mathcal{O}(t') = \mathcal{O}(t) + q_s \cdot (k \cdot \texttt{PRF} + (t+2)\texttt{FHE.Eval(PRF)})$.

- *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 $\texttt{HORS}$ public key. This implies that $(sk_i^w = \overline{sk}, PK_i^w = \overline{PK})$ holds. Since $\texttt{HORS.Kg}(.)$ generates the secret keys random uniformly (Definition 1), 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* $\texttt{INF-HORS}$ *scheme provides PQ promises.*

Proof: Based on the preceding formal security analysis and the incorporation of cryptographic primitives such as FHE, PRF, and hash functions, the $\texttt{INF-HORS}$ 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 [15], are constructed upon lattice-based hard problems like General-LWE assumptions, thereby reinforcing the assurance of PQ promises.

6. Performance Analysis and Comparison

In this section, we give a detailed performance analysis of $\texttt{INF-HORS}$ and compare it with its counterparts.

6.1. Evaluation Metrics and Experimental Setup

Evaluation Metrics: Our analysis evaluates $\texttt{INF-HORS}$ and its analogous counterparts, with a main focus on the signer efficiency that includes: **(i)** private key and signature sizes which translates into 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 [65]).

Parameter Selection: Our system-wide parameters are $I = (I_{\texttt{HORS}}, I_{\texttt{FHE}}, I_{\texttt{DM}})$. We choose $I_{\texttt{HORS}} \leftarrow (k = 16, t = 1024)$, where SHA-256 and $\texttt{DM}$ are used as H and f (i.e., $\texttt{OWF}$), respectively. In $I_{\texttt{DM}}$, we choose AES-128 with Galois/Counter Mode (GCM) as our $\texttt{PRF}$. In $I_{\texttt{FHE}}$, 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 that empowers us 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 $\texttt{INF-HORS}$ on both commodity hardware and two selected embedded devices.

- *Commodity Hardware:* is a resourceful desktop equipped with an Intel i9-9900K@3.6GHz processor and 64GB of RAM.
- *Embedded device:* We selected an 8-bit ATxmega128A1 microcontroller due to assess the efficiency of $\texttt{INF-HORS}$ on embedded IoT devices. It is equipped with 128KB flash memory, 2KB RAM, 8KB EEPROM, with 32MHz as clock frequency.

Software Configuration: For the commodity hardware, we utilized the following libraries (i) OpenSSL[3] to implement SHA-256 (ii) HElib[4] to implement $\texttt{FHE}$ functionalities (*e.g.*,

[3] https://github.com/openssl/openssl

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

Table 2
Performance comparison of INF-HORS and its counterparts on commodity hardware

Scheme	Signing Time (μs)	Private Key	Signature Size	Verification Time (μs)	Verifier Storage Pub Key	Verifier Storage Cert.	Verifier Storage Tot. (2^{20} users) (GB)	Post-Quantum Promise	Sampling Operations	Simple Code Base
ECDSA [36]	16.98	0.06	0.06	46.41	0.09	0.06	0.16	×	×	×
Ed25519 [11]	16.39	0.06	0.06	39.75	0.09	0.06	0.16	×	×	×
BLISS-I [24]	244.97	2.00	5.6	25.21	7.00	5.6	12.6	✓	✓	×
Dilithium-II [25]	93.76	2.29	2.36	18.73	1.28	2.36	3.75	✓	✓	×
Falcon-512 [26]	184.74	1.29	0.65	32.16	0.88	0.65	1.53	✓	✓	×
SPHINCS+ [12]	5,441.58	0.13	32.63	549.63	0.06	32.63	32.69	✓	×	×
XMSS$^{\text{MT}}$[35]	10,682.35	3.11	2.61	2,098.84	0.75	2.61	3.36	✓	×	×
INF-HORS	**4.81**	**0.02**	**0.25**	Ver (Online) **1.91 *s*** PKConstr (Offline) **41.22 *s***	**9.42 MB**			✓	×	✓

The private/public key, signature, and certificate sizes are in KB. INF-HORS and NIST PQC candidates use architecture-specific optimizations (i.e., AESNI, AVX2 instructions). For XMSS$^{\text{MT}}$, we choose the XMSST_MT_SHA2_20_256 variant. For SPHINCS+, 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.

Table 3
Performance comparison of INF-HORS and its counterparts at the signer side on embedded devices

Scheme	Signing (cycles)	Secret Key (KB)	Signature Size (KB)	Post-Quantum Promise	Rejection Sampling	Ease of Implementation
ECDSA [36]	34,903,000	0.06	0.06	×	×	×
Ed25519 [11]	22,688,583	0.06	0.06	×	×	×
BLISS-I [24]	10,537,981	2	5.6	✓	✓	×
INF-HORS	**514,788**	**0.02**	**0.25**	✓	×	✓

The counterpart selection covers existent the most efficient conventional (ECDSA, Ed25519) and PQ-secure (BLISS) with an available benchmark on the selected 8-bit AVR MCU.

evaluation and comparison under encryption[5]). (iii) DM is implemented using the hardware-optimized AES-NI [32]. The Raspberry Pi 4.0 supports the same cryptographic libraries as in commodity hardware except the AES-NI optimized implementation. For the 8-bit AVR device, we employed the AVR cryptographic library[6] to implement AES-128 and SHA-256. This library offers an optimized assembly implementation, resulting in mimimal cycles for evaluating hashing and PRF calls.

Selection Rationale of Counteparts: 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 *(i) lattice-based*: the NIST PQC standards Dilithium-II [25] and Falcon-512 [26]. They are considered the most prominent lattice-based signatures, with balanced efficiency between key sizes and signing efficiency. We also selected BLISS-II because it is the only lattice-based signature with a benchmark on an 8-bit AVR MCU [54]. *(ii) hash-based*: generally suffer from an expensive signing cost with larger key sizes. We selected the NIST PQC standard SPHINCS+ [12], a stateless signature scheme. We also selected XMSS$^{\text{MT}}$ [35] as a standard stateful hash-based signature with forward security. To our knowledge, there is no hash-based signature with a benchmark on 8-bit AVR MCUs. *(iii) multivariate-based*: are known to be computationally efficient in terms of signing and verification with small signature and public key sizes. However, they generally suffer from large private key sizes, resulting in high memory usage and frequent access. This limitation might be problematic when deployed on highly constrained 8-bit devices with 128KB of static flash memory. There exist numerous multivariate-based digital signatures that have been proposed (e.g., [40, 63, 64]). We identified HiMQ-3$^{\text{Big}}$ [64] that achieve a high signing efficiency on an 8-bit AVR ATxmega384C3. However, we observed a high memory usage that includes the private key size and code size, occupying 72.38% of the flash read-only memory of our target ATxmega128A1. Therefore, we omit it from our performance analysis due to the high memory usage. *(iv) 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 [36] and Ed25519 [11]. Other conventional (e.g., pairing-based [14]) digital signatures incur expensive operations during signature generation and are therefore not practical for resource-limited IoT devices.

[5] https://github.com/iliailia/comparison-circuit-over-fq/tree/master

[6] https://github.com/cantora/avr-crypto-lib

6.2. Performance on Signer

We present the performance comparison on commodity hardware and the embedded device in Tables 2 and 3, respectively.

• *Memory Usage:* INF-HORS achieves the lowest memory usage by having the smallest private key size among its counterparts. For example, the private key of INF-HORS is 3× and 114× smaller than that of the conventional signer-efficient Ed25519 and PQ-secure Dilithium standards, respectively. The private key is 22× smaller than that of the most efficient lattice-based counterpart, BLISS-I [24], respectively. It is without incurring large code size and expensive costly sampling operations that may result in side-channel attacks [67]. Notably, INF-HORS consumes less memory than its most signer-efficient and PQ-secure counterpart, HiMQ-3^{Big}[64], by having a significantly smaller private key size. The cryptographic storage (including the code size) of HiMQ-3^{Big} utilizes 72.38% of the overall flash memory size of an 8-bit AVR ATxmega128A1, whereas INF-HORS utilizes only 2.8%. We argue that the cryptographic data should occupy minimal space, particularly in resource-limited devices (e.g., pacemakers, medical implants). 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 the high memory cryptographic usage. It is noteworthy that we do not assess the impact of memory access on the battery lifetime of the embedded device. We foresee a high energy usage of multivariate-based signatures compared to that of INF-HORS, considering the high cost of both memory and stack usage.

• *Bandwidth Overhead:* INF-HORS boasts a compact signature size that is 9.4× and 2.6× smaller than the NIST PQC standards, Dilithium-II and Falcon-512, respectively. The signature size of INF-HORS 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), INF-HORS exhibits the fastest signing time and the lowest signer storage overhead. It is 10× and 43× faster than the NIST PQC standards, Dilithium-II and Falcon-512, respectively. The computational performance advantages at the signer of INF-HORS become even more apparent on embedded devices. Based on 8-bit AVR MCU results in Table 3, the signing time of INF-HORS 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 INF-HORS, in Figure 2, we profiled the battery depletion with respect to the signing operations. Specifically, we plot the battery status while solely performing signature generation operations on the 8-bit ATxmega128A1 MCU. Remind that, to the best of our knowledge, none of the selected NIST PQC signatures have an open-source implementation available on such resource-limited devices (i.e., 8-bit microcontrollers). The most prominent PQ alternatives with a reported performance on this platform are HiMQ-3^{big} [64] and BLISS-I [24]. We also included the most efficient ECC-based alternative Ed25519 and the widely-used ECDSA in our energy comparison to assess INF-HORS performance with respect to (pre-quantum) conventional schemes. Figure 2 showcases that INF-HORS offer the longest battery lifetime when only the cryptographic computation is considered. Hence, we confirm that INF-HORS is the most suitable signature scheme for highly resource-constrained IoT devices. We foresee that this finding will benefit implantable medical devices (e.g., a cardiac monitor) when device replacement requires an implant procedure.

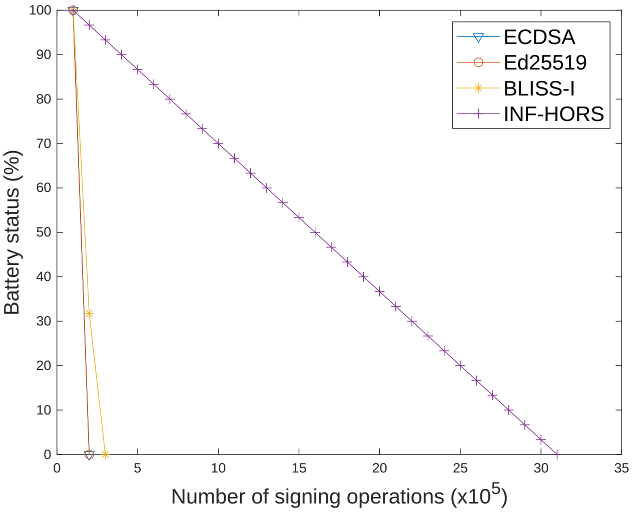

Figure 2: Impact of signing on battery lifetime for AVR ATxmega128A1

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 [33], which hinders its deployment on 8-bit microcontrollers. INF-HORS 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 INF-HORS 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.

6.3. Performance on Verifier

While INF-HORS is a signer-optimal scheme, we also introduced strategies to minimize the verifier computational and storage overhead. As explained in Section 4.1, 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 $k \times \texttt{PRF}(.)$, $k \times \texttt{FHE.Enc}(.)$, and $k \times \texttt{FHE.Eval}(.)$ 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.

Offline Public Key Construction: The main computational bottleneck of INF-HORS is the offline phase. In our tests, the average cost of a single homomorphic AES evaluation per block is 2.29 seconds. Hence, the overall cost of generating k public key components is 41.22 seconds. We note that the offline computational overhead can be significantly reduced with parallelizations. For example, since each component of the HORS public key can be generated independently, they can be assigned to different threads or computing units. Moreover, as discussed in Section 4.1, encrypted AES evaluations via BGV are highly parallelizable, which is one of our reasons to opt for AES as our DM building block.

Verifier Storage Overhead: The total size of the master public key $\boldsymbol{PK}$ with the expansion per block evaluation is around 9.42 MB. If only a single signer is considered, the size of $\boldsymbol{PK}$ is much larger than that of its counterparts. However, INF-HORS enables a verifier to construct public keys for *any valid* signer $\boldsymbol{ID}_i$ of any state. This unique property permits INF-HORS 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 INF-HORS, while it is around 1.52 GB and 3.74 GB for Falcon-512 and Dilithium-II, respectively. The total storage advantage increases with a growing number of signers.

7. Conclusion

In conclusion, the burgeoning field of Medical Internet of Things (MIoT) demands robust security measures to safeguard the sensitive medical data transmitted by resource-limited embedded devices. While conventional digital signatures offer authentication and integrity, their vulnerability to emerging quantum computers poses a long-term security risk. Current post-quantum cryptography (PQC) standards, though promising, impose heavy overhead unsuitable for battery-limited MIoT devices. Addressing this gap, this paper introduces INFinity∓HORS (INF-HORS), a lightweight post-quantum digital signature optimized for signer efficiency and minimal architectural assumptions. Notably, INF-HORSoutperforms existing PQC standards in computational efficiency while maintaining compact memory and signature footprints, as evidenced by experimental results on an 8-bit microcontroller. Moreover, INF-HORSmitigates security risks by eschewing additional system or architectural assumptions, thus proving ideal for enhancing MIoT security without compromising device resources or imposing stringent security dependencies.

8. Acknowledgment

This work is supported by Cisco Research Award (220159) and National Science Foundation CAREER Award CNS (1917627).

References

[1] Adamson, P.B., 2009. Pathophysiology of the transition from chronic compensated and acute decompensated heart failure: new insights from continuous monitoring devices. Current heart failure reports 6, 287–292.

[2] Adeli, M., Bagheri, N., Maimani, H.R., Kumari, S., Rodrigues, J.J., 2023. A post-quantum compliant authentication scheme for iot healthcare systems. IEEE Internet of Things Journal .

[3] Ahad, A., Tahir, M., Aman Sheikh, M., Ahmed, K.I., Mughees, A., Numani, A., 2020. Technologies trend towards 5G network for smart health-care using iot: A review. Sensors 20, 4047.

[4] Aliasgari, M., Black, M., Yadav, N., 2018. Security vulnerabilities in mobile health applications, in: 2018 IEEE Conference on application, information and network security (AINS), IEEE. pp. 21–26.

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

[6] Aumasson, J.P., 2017. The impact of quantum computing on cryptography. Computer Fraud & Security 2017, 8–11.

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

[8] Behnia, R., Ozmen, M.O., Yavuz, A.A., 2019. Aris: authentication for real-time iot systems, in: ICC 2019-2019 IEEE International Conference on Communications (ICC), IEEE. pp. 1–6.

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

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

[11] Bernstein, D.J., Duif, N., Lange, T., Schwabe, P., Yang, B.Y., 2012. High-speed high-security signatures. Journal of cryptographic engineering 2, 77–89.

[12] Bernstein, D.J., Hülsing, A., Kölbl, S., Niederhagen, R., Rijneveld, J., Schwabe, P., 2019. The SPHINCS+ signature framework, in: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, pp. 2129–2146.

[13] Beullens, W., 2021. Mayo: practical post-quantum signatures from oil-and-vinegar maps, in: International Conference on Selected Areas in Cryptography, Springer. pp. 355–376.

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

[15] Brakerski, Z., Gentry, C., Vaikuntanathan, V., 2014. (leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT) 6, 1–36.

[16] Camara, C., Peris-Lopez, P., De Fuentes, J.M., Marchal, S., 2020. Access control for implantable medical devices. IEEE Transactions on Emerging Topics in Computing 9, 1126–1138.

[17] Camara, C., Peris-Lopez, P., Tapiador, J.E., 2015. Security and privacy issues in implantable medical devices: A comprehensive survey. Journal of biomedical informatics 55, 272–289.

[18] Chen, X., He, D., Khan, M.K., Luo, M., Peng, C., 2022. A secure certificateless signcryption scheme without pairing for internet of medical things. IEEE Internet of Things Journal 10, 9136–9147.

[19] Cheng, C., Lu, R., Petzoldt, A., Takagi, T., 2017. Securing the internet of things in a quantum world. IEEE Communications Magazine 55, 116–120.

[20] Chillotti, I., Gama, N., Georgieva, M., Izabachène, M., 2020. Tfhe: fast fully homomorphic encryption over the torus. Journal of Cryptology 33, 34–91.

[21] Costello, C., Longa, P., 2016. SchnorrQ: Schnorr signatures on fourQ. MSR Tech Report, 2016 .

[22] Darzi, S., Ahmadi, K., Aghapour, S., Yavuz, A.A., Kermani, M.M., 2023. Envisioning the future of cyber security in post-quantum era: A survey on pq standardization, applications, challenges and opportunities. arXiv preprint arXiv:2310.12037 .

[23] Darzi, S., Yavuz, A.A., 2024. Pqc meets ml or ai: Exploring the synergy of machine learning and post-quantum cryptography. Authorea Preprints .

[24] Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V., 2013. Lattice signatures and bimodal gaussians, in: Cryptology Conf., pp. 40–56.

[25] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D., 2018. Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems , 238–268.

[26] Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z., 2018. Falcon: Fast-fourier lattice-based compact signatures over NTRU. Submission to the NIST's post-quantum cryptography standardization process .

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

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

[29] Halperin, D., Heydt-Benjamin, T.S., Fu, K., Kohno, T., Maisel, W.H., 2008. Security and privacy for implantable medical devices. IEEE pervasive computing 7, 30–39.

[30] Hashimoto, Y., Takagi, T., Sakurai, K., 2011. 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. pp. 1–18.

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

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

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

[34] Hülsing, A., Rausch, L., Buchmann, J., 2013. Optimal parameters for XMSS MT, in: International conference on availability, reliability, and security, pp. 194–208.

[35] Hülsing, A., Rausch, L., Buchmann, J., 2017. Optimal parameters for XMSSˆ MT. Cryptology ePrint Archive, Paper 2017/966 .

[36] Johnson, D., Menezes, A., Vanstone, S., 2001. The elliptic curve digital signature algorithm (ECDSA). International journal of information security 1, 36–63.

[37] Joung, Y.H., 2013. Development of implantable medical devices: from an engineering perspective. International neurourology journal 17, 98.

[38] Karabulut, E., Aysu, A., 2021. Falcon down: Breaking falcon post-quantum signature scheme through side-channel attacks, in: 2021 58th ACM/IEEE Design Automation Conference (DAC), pp. 691–696.

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

[40] Li, Q., He, D., Chen, Y., Wen, J., Yang, Z., 2024. An efficient quantum-resistant undeniable signature protocol for the e-voting system. Journal of Information Security and Applications 81, 103714.

[41] Liu, H., Han, D., Cui, M., Li, K.C., Souri, A., Shojafar, M., 2023. Idenmultisig: Identity-based decentralized multi-signature in internet of things. IEEE Transactions on Computational Social Systems .

[42] Martínez, A.L., Pérez, M.G., Ruiz-Martínez, A., 2023. A comprehensive model for securing sensitive patient data in a clinical scenario. IEEE Access 11, 137083–137098.

[43] McGrew, D., Curcio, M., Fluhrer, S., 2019. Leighton-Micali Hash-Based Signatures. RFC 8554.

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

[45] Merkle, R.C., 1989. A certified digital signature, in: Conference on the Theory and Application of Cryptology, Springer. pp. 218–238.

[46] Mudgerikar, A., Bertino, E., 2021. Iot attacks and malware. Cyber Security Meets Machine Learning , 1–25.

[47] Nguyen, D.C., Ding, M., Pathirana, P.N., Seneviratne, A., Li, J., Niyato, D., Dobre, O., Poor, H.V., 2021. 6G internet of things: A comprehensive survey. IEEE Internet of Things Journal .

[48] NIST, . Post-Quantum Cryptography Standardization. `https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization`. Accessed: May 23, 2024.

[49] Nouma, S.E., , Yavuz, A.A., 2023. Post-quantum forward-secure signatures with hardware-support for internet of things, IEEE. p. 1–7.

[50] Nouma, S.E., Yavuz, A.A., 2023. 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, pp. 340–353.

[51] Nouma, S.E., Yavuz, A.A., 2024a. Trustworthy and efficient digital twins in post-quantum era with hybrid hardware-assisted signatures. ACM Transactions on Multimedia Computing, Communications and Applications 20, 1–30.

[52] Nouma, S.E., Yavuz, A.A., 2024b. Trustworthy and efficient digital twins in post-quantum era with hybrid hardware-assisted signatures. ACM Trans. Multimedia Comput. Commun. Appl. 20.

[53] Ouyang, W., Wang, Q., Wang, W., Lin, J., He, Y., 2021. Scb: Flexible and efficient asymmetric computations utilizing symmetric cryptosystems implemented with intel sgx, in: 2021 IEEE International Performance, Computing, and Communications Conference (IPCCC), IEEE. pp. 1–8.

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

[55] Pornin, T., 2013. Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA). RFC 6979.

[56] Pradhan, M., Noll, J., 2020. Security, privacy, and dependability evaluation in verification and validation life cycles for military iot systems. IEEE Communications Magazine 58, 14–20.

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

[58] Reyzin, L., Reyzin, N., 2002. Better than BiBa: Short one-time signatures with fast signing and verifying, in: Australasian Conference on Information Security and Privacy, pp. 144–153.

[59] Rieback, M.R., Crispo, B., Tanenbaum, A.S., 2005. Rfid guardian: A battery-powered mobile device for rfid privacy management, in: Australasian Conference on Information Security and Privacy, Springer. pp. 184–194.

[60] Sametinger, J., Rozenblit, J., Lysecky, R., Ott, P., 2015. Security challenges for medical devices. Communications of the ACM 58, 74–82.

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

[62] Sehgal, A., Perelman, V., Kuryla, S., Schonwalder, J., 2012. Management of resource constrained devices in the internet of things. IEEE Communications Magazine 50, 144–149.

[63] Shaw, S., Dutta, R., 2022. Post-quantum secure identity-based signature achieving forward secrecy. Journal of Information Security and Applications 69, 103275.

[64] Shim, K.A., Park, C.M., Koo, N., Seo, H., 2020. A high-speed public-key signature scheme for 8-b iot-constrained devices. IEEE Internet of Things Journal 7, 3663–3677.

[65] Shor, P.W., 1999. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review 41, 303–332.

[66] Srivastava, V., Debnath, S.K., Tiwari, S.K., Singh, H., 2023. On the security of multivariate-based ring signature and other related primitives. Journal of Information Security and Applications 74, 103474.

[67] Tibouchi, M., Wallet, A., 2021. One bit is all it takes: a devastating timing attack on BLISS's non-constant time sign flips. Journal of Mathematical Cryptology 15, 131–142.

[68] Verma, G.K., Singh, B., Kumar, N., Chamola, V., 2019. Cb-cas: Certificate-based efficient signature scheme with compact aggregation for industrial internet of things environment. IEEE Internet of Things Journal 7, 2563–2572.

[69] Virani, S.S., Alonso, A., Benjamin, E.J., Bittencourt, M.S., Callaway, C.W., Carson, A.P., Chamberlain, A.M., Chang, A.R., Cheng, S., Delling, F.N., et al., 2020. Heart disease and stroke statistics—2020 update: a report from the american heart association. Circulation 141, e139–e596.

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

[71] Wu, Y., Zhang, K., Zhang, Y., 2021. Digital twin networks: A survey. IEEE Internet of Things Journal 8, 13789–13804.

[72] Yavuz, A.A., 2013. Eta: efficient and tiny and authentication for heterogeneous wireless systems, in: Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks, pp. 67–72.

[73] Yavuz, A.A., Ozmen, M.O., 2019. Ultra lightweight multiple-time digital signature for the internet of things devices. IEEE Transactions on Services Computing , 215–227.

[74] Yavuz, A.A., Sedghighadikolaei, K., Darzi, S., Nouma, S.E., 2023. 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. pp. 267–276.

[75] Yu, Y., Xie, X., 2021. Privacy-preserving computation in the post-quantum era. National Science Review 8. doi:10.1093/nsr/nwab115. nwab115.

[76] Zhang, G., Liao, Y., Fan, Y., Liang, Y., 2020. Security analysis of an identity-based signature from factorization problem. IEEE Access 8, 23277–23283.

[77] Zhang, M., Raghunathan, A., Jha, N.K., 2013. Medmon: Securing medical devices through wireless monitoring and anomaly detection. IEEE Transactions on Biomedical circuits and Systems 7, 871–881.

[78] Zile, M.R., Bennett, T.D., St. John Sutton, M., Cho, Y.K., Adamson, P.B., Aaron, M.F., Aranda Jr, J.M., Abraham, W.T., Smart, F.W., Stevenson, L.W., et al., 2008. Transition from chronic compensated to acute decompensated heart failure: pathophysiological insights obtained from continuous monitoring of intracardiac pressures. Circulation 118, 1433–1441.