Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.085904

ARTICLE

Side-Channel-Resistant Post-Quantum Digital Signatures with Verkle Trees, Lattice-Based Vector Commitments, and Quantum True Random Number Generators

Maksim Iavich[1], **Nursulu Kapalova**[2] **and Kunbolat Algazy**[2,*]

[1]Department of Computer Science, Caucasus University, Tbilisi, Georgia
[2]Information Security Laboratory, Institute of Information and Computational Technologies, Almaty, Kazakhstan

*Corresponding Author: Kunbolat Algazy. Email: kunbolat@mail.ru

Received: 20 May 2026; Accepted: 14 July 2026; Published: 13 August 2026

ABSTRACT: Lattice-based post-quantum cryptographic standards such as Module-Lattice Key Encapsulation Mechanism (ML-KEM) and Module-Lattice-Based Digital Signature Algorithm (ML-DSA) have demonstrated documented susceptibility to power-based side-channel attacks even when protected by higher-order arithmetic masking. Concurrently, hash-based and Verkle-tree digital signature schemes lack a systematic analysis of their physical-layer attack surface. This paper closes both gaps by introducing a Verkle-tree digital signature scheme incorporating multiple complementary countermeasures: (i) arithmetic masking of lattice-based Short Integer Solution (SIS) vector commitments, (ii) a counter-mode deterministic random bit generator (CTR_DRBG) seeded by a hardware quantum random number generator (QRNG), and (iii) an implementation framework experimentally validated on ChipWhisperer-Nano and ChipWhisperer-Husky embedded platforms. We leverage ID Quantique Quantis PCIe, ID Quantique Quantum Appliance, and CryptoLabs USB QRNG modules as entropy sources; their certified output distributions are characterized and formally incorporated into the key-generation security proof. We analyze the susceptibility of existing post-quantum lattice schemes to correlation power analysis (CPA), higher-order CPA (HOCPA), and single-trace soft-analytical attacks (SASCA), and we qualitatively discuss why deep-learning side-channel attacks (DLSCA) face additional structural challenges against our Verkle-SIS construction. We demonstrate that the structural properties of Verkle trees, combined with on-demand key generation and QRNG-seeded mask refreshing, measurably reduce the physical attack surface compared with standard NTT-based polynomial multiplication targets. Full Existential Unforgeability under Chosen Message Attack (EUF-CMA) security is proven in the quantum random oracle model (QROM) under the SIS hardness assumption. Our ChipWhisperer measurements confirm that first- through third-order CPA attacks against the SIS commitment step require at least 23×10^3 traces to exceed a 50% success rate, compared with 700–2400 traces sufficient to break equivalently masked Dilithium implementations. While QRNG seeding provides measurable improvements in mask-refreshing quality over classical hardware entropy sources, it is one component of a layered protection strategy and does not by itself guarantee side-channel resistance.

KEYWORDS: Post-quantum cryptography; side-channel analysis; Verkle tree; lattice-based vector commitments; quantum random number generator; ChipWhisperer; masking; correlation power analysis; CTR_DRBG; EUF-CMA

1 Introduction

The imminent threat of cryptographically relevant quantum computers has driven the standardization of post-quantum cryptographic (PQC) primitives by the National Institute of Standards and Technology (NIST). Among the inaugural standards, the ML-KEM (formerly Crystals-Kyber) and the ML-DSA

(formerly Crystals-Dilithium) occupy central roles [1]. Their security rests on the assumed hardness of the Module Learning With Errors (M-LWE) and Module Short Integer Solution (M-SIS) problems against polynomial-time quantum algorithms. However, theoretical security proofs offer no guarantees against adversaries who monitor the physical emanations of a computing device during cryptographic operations. Side-channel analysis (SCA) exploits correlations between secret-dependent intermediate values and observable physical quantities such as instantaneous power consumption or electromagnetic (EM) radiation [2].

Recent literature has firmly established that Kyber and Dilithium implementations remain vulnerable to SCA even when protected by higher-order arithmetic masking. Tosun et al. [3] demonstrated complete secret-key recovery from second-order masked Dilithium and Kyber with 700 and 2200 power traces, respectively, by exploiting the Hamming-weight leakage of the Number Theoretic Transform (NTT)-based base multiplication. Primas et al. [4] showed that a single Electromagnetic (EM) trace suffices for full decryption-key recovery from masked Ring Learning with Errors (RLWE) implementations when belief propagation is applied to the NTT factor graph. Iavich and Kuchukhidze [5] surveyed deep-learning side-channel attacks capable of breaking fifth-order masked Kyber by exploiting leakage in the message re-encoding step during decapsulation.

Verkle-tree digital signature schemes [6,7] have attracted attention as an alternative paradigm combining hash-based one-time signature keys with vector-commitment-based authentication paths. Their structural properties are fundamentally different from NTT-based polynomial multiplications, which constitute the primary SCA target in lattice-based schemes. Despite this promising feature, no prior work has formally analyzed the SCA resistance of Verkle-tree schemes or proposed countermeasures grounded in measured physical leakage data.

This paper addresses these open problems with the following contributions.

1. First architectural SCA countermeasure for Verkle signatures. Prior Verkle-tree schemes [6,7] provided no side-channel analysis. We provide the first systematic SCA evaluation of Verkle-based signatures and the first masked implementation of SIS vector commitments.
2. Novel integration of QRNG-seeded CTR_DRBG into Verkle key generation. While QRNGs and CTR_DRBG are individually standard, their combination with forward-secret Winternitz One-Time Signature (W-OTS) key derivation (Eq. (7)) creates a novel countermeasure against retrospective SCA: even if an attacker recovers the current seed from a power trace, previous signing keys remain irrecoverable.
3. Quantitative comparison of SIS vs. NTT leakage. No prior work has directly compared the CPA/HOCPA trace complexity of SIS-based commitments vs. NTT-based polynomial multiplication under identical masking orders. We provide the first head-to-head experimental comparison (Table 1).
4. Formal incorporation of measured QRNG min-entropy into QROM proof. Previous proofs assume perfect randomness. We measure $H_{\min}$ from three certified QRNGs (Table 2) and embed the conservative bound $\gamma = 255$ into the security reduction (Section 5.1).
5. Empirical validation of seed-quality impact on SCA. We experimentally demonstrate (Section 6.4) that low-entropy seeding (32-bit LCG) collapses masking resistance—a result that has implications far beyond our scheme.

Table 2 compares our scheme with prior Verkle-tree and hash-based signature schemes across key security and performance metrics.

Table 1: Minimum number of power traces N required for full secret-key recovery under HOCPA against masked Kyber768 and Dilithium3, following post-processing with lattice reduction (BKZ-20/BKZ-50). Data from [3].

Algorithm	$d = 1$	$d = 2$	$d = 3$	$d = 4$	Prediction Function
Kyber768	21	380	2200	14,500	$\|x\|$, sin, cos, − sin
Dilithium3	25	190	700	2400	$\|x\|$, sin, cos, − sin

Table 2: Comparison of signature schemes—side-channel analysis, masking, QRNG integration, and forward secrecy.

Scheme	SCA Analysis	Masking	QRNG Integration	Forward Secrecy	Signature Size (Approx.)
Iavich & Kapalova [6]	None	None	None (PRNG)	No	~8 KB
Iavich et al. [7]	None	None	None	No	~10 KB
XMSS (hash-based)	None	None	None	Yes (state)	~2.5 KB
This work	Yes (CPA/HOCPA)	$d = 3$	QRNG ($H_{\min} \geq 0.998$)	Yes	~9.4 KB

The remainder of this paper is organized as follows. Section 2 provides background on Verkle-tree signatures, SIS vector commitments, and the side-channel attack taxonomy. Section 3 reviews relevant SCA attacks on PQC lattice schemes. Section 4 describes the proposed scheme, QRNG integration, and masking strategy. Section 5 presents the formal security analysis. Section 6 reports ChipWhisperer experimental results. Section 7 discusses practical implications and mitigations. Section 8 concludes.

2 Background

2.1 Verkle Trees and Hash-Based Signatures

A Verkle tree is a k-ary authenticated data structure that replaces the cryptographic hash function used in Merkle tree internal nodes with a vector commitment (VC) scheme [8]. Given a branching factor k and tree height h, the structure can authenticate k^h leaf values. The distinguishing feature is that membership proofs scale as $O(\log_k n)$ in the number of leaves n, as opposed to $O(\log_2 n)$ for binary Merkle trees. For $k = 16$, this yields a 60% reduction in proof size.

The Merkle signature scheme (MSS) [9] organizes 2^H one-time key pairs $(X_j, Y_j), 0 \leq j < 2^H$, into a binary hash tree whose root acts as the public key. Each signature on message M is a tuple $(s, sign, Y_s, auth - path)$, where sign is generated under the s-th Winternitz OTS (W-OTS) key X_s. Verification reconstructs the root from $g(Y_s)$ and the authentication path. Replacing hash-based internal nodes with VC-based ones transforms MSS into a Verkle signature scheme, with proof sizes reduced accordingly.

Compared to NIST-standardized ML-DSA [1], Verkle-tree signatures avoid NTT-based polynomial multiplication entirely in their signing path. This structural difference is central to our side-channel argument in Section 3.

2.2 Lattice-Based Vector Commitments

A vector commitment (VC) scheme over message space $\mathcal{M} = \mathbb{Z}$ commits to a length-d vector $m \in \mathcal{M}^d$ producing a commitment $c \in \mathbb{Z}_q^n$ and, for any index i, a proof π_i enabling a verifier to confirm that m_i is

the i-th committed value without revealing other entries. Position-binding ensures that no polynomial-time adversary can open c at position i to two different values simultaneously.

The SIS-based construction [10] used in this work relies on a randomly chosen matrix $A \in \mathbb{Z}_q^{n\times m}$, a gadget matrix $G \in \mathbb{Z}_q^{n\times w}$, and encoding matrices $H_i \in \mathbb{Z}_q^{n\times n}$, for $i \in [d+1]$. For a prime q and the choice $H_i = i \cdot I_n$, the commitment of $m \in \mathbb{Z}_q^d$ is

$$c = \sum_{j=0}^{d-1} U_j \cdot m_j \,(mod\ q) \tag{1}$$

where,

$U = [U_0, \ldots, U_{d-1}] \in \mathbb{Z}_q^{n\times d}$.

The proof for position i is

$$\pi_i = R_i \cdot m_i + \sum_{\substack{j=0 \\ j\neq i}}^{d-1} R_{ij} \cdot m_j \in \mathbb{Z}^m \tag{2}$$

where the short matrices $R_{ij} \in \mathbb{Z}^m$ satisfy $A \cdot R_{ij} \equiv U_j (mod\ q)$.

Security reduces to the hardness of the Short Integer Solution (SIS) problem: given $A \in \mathbb{Z}_q^{n\times m}$, find a non-zero vector $v \in \mathbb{Z}^m$ such that $Av \equiv 0 (mod\ q)$ and $\| v \| \leq \beta$, for a norm bound β.

For concreteness, we instantiate the SIS-based commitment with parameters $n = 256$, $m = 1024$, $q = 2^{23}$, and $\beta = 2\sqrt{n}$, following the recommendations of de Castro and Peikert [11] for achieving 128-bit post-quantum security.

Under these parameters, the hardness of the SIS problem $SIS_{n,m,q,\beta}$ is defined as finding a non-zero vector $x \in \mathbb{Z}^m \setminus \{0\}$ such that $Ax \equiv 0 (mod\ q)$, while satisfying the norm bound $\| x \| \leq \beta$, where $A \in \mathbb{Z}_q^{n\times m}$ is a uniformly random matrix.

For these parameter choices, the best known quantum attacks on lattice problems—such as sieving algorithms with extreme pruning—require computational effort on the order of 2^{128} operations, thereby providing a security level consistent with NIST post-quantum standards [10].

2.3 Pseudo-Random and True Random Number Generators in Cryptography

Cryptographic key material requires entropy that is both uniformly distributed and unpredictable to all computationally bounded adversaries. Classical pseudo-random number generators (PRNGs) can be seeded with entropy from hardware sources, but the quality of that entropy limits forward and backward security. The NIST CTR_DRBG [12] generates random bits by encrypting a monotonically incremented counter with Advanced Encryption Standard (AES), initialized by a seed of n bits. Its security is well characterized in SP 800-90A for seeds of 256 bits, providing 2^{128} security even against Grover-accelerated quantum exhaustive search on the AES key [12].

Quantum random number generators (QRNGs) harvest entropy from quantum-mechanical processes—typically vacuum fluctuations or single-photon detection—that are provably non-deterministic even to a computationally unbounded observer [13]. The three hardware platforms employed in our laboratory are:

- ID Quantique Quantis PCIe: A PCI-Express add-in card based on quantum shot noise of single-photon detection. Certified to AIS-31 (Class P2), operating at up to 240 Mbit/s raw throughput with on-board randomness extraction.
- ID Quantique Quantum Appliance (rack-mounted): A network-attached QRNG appliance delivering DRBG-post-processed entropy via REST (Representational State Transfer) or KMIP (Key Management Interoperability Protocol) interfaces, certified to FIPS 140-2 Level 3 and Common Criteria EAL4+.
- CryptoLabs USB QRNG: A compact USB-form-factor module based on optical quantum noise, delivering 10 Mbit/s certified entropy output with SHA-3 post-processing. Suitable for portable and IoT deployment scenarios.

All three devices were characterized using the NIST SP 800-90B test suite. Min-entropy estimates $H_{\min}$ were computed per the Independent and Identically Distributed (IID) track. Results are summarized in Section 6.

2.4 Side-Channel Analysis: Taxonomy and Metrics

Side-channel attacks (SCAs) exploit unintended information leakage via physical side channels. The primary channels of relevance to embedded cryptographic implementations are:

- Power consumption (simple power analysis, SPA; differential power analysis, DPA; correlation power analysis, CPA).
- Electromagnetic (EM) emanations.
- Timing.

The standard leakage model for software implementations assumes that the instantaneous power consumption L is a noisy function of the Hamming weight W of processed data, given by

$$L = W(X) + N(\mu, \sigma) \tag{3}$$

where X is a secret-dependent intermediate variable and $N(\mu, \sigma)$ is additive Gaussian noise. The attacker correlates hypothetical intermediate values with observed traces using Pearson's correlation coefficient (CPA [14]) or, for masked implementations, a combination function such as the mean-free product (HOCPA [15]).

For a d-th order masked implementation, the intermediate variable is shared as

$$X = \sum_{i=0}^{d-1} X^{(i)} \pmod q .$$

The d-th order optimal prediction function is defined as

$$f_{\text{opt}}^{(d)}(x) = \mathbb{E}\left[C\left(\{L'(X^{(i)})\}_{i=0}^{d-1}\right) | X = x\right] \tag{4}$$

where, $C(\cdot)$ denotes the mean-free product combination function, and the expectation is taken over uniformly random shares $X^{(0)}, \ldots, X^{(d-2)}$.

Tosun et al. [3] showed that when signed modular arithmetic is used, the optimal prediction functions admit closed-form expressions:

$$f_{\text{opt}}^{(2)}(x) = \mid x \mid, f_{\text{opt}}^{(3)}(x) = \sin\left(\frac{2\pi x}{q}\right), f_{\text{opt}}^{(4)}(x) = \cos\left(\frac{2\pi x}{q}\right)$$

enabling efficient higher-order side-channel attacks.

2.5 Related Work

This subsection situates our contribution within the broader literature.

NIST has standardized three post-quantum signature families: lattice-based ML-DSA (Dilithium) and Falcon, hash-based SPHINCS+ (Stateless Practical Hash-based INCredible Signature), and stateful XMSS/LMS [1]. ML-DSA relies on NTT-based polynomial multiplication, which we identify as a primary side-channel target.

Tosun et al. [3] demonstrated higher-order CPA against masked Dilithium and Kyber, recovering keys with 700–2200 traces. Primas et al. [4] showed single-trace belief propagation attacks on RLWE decryption. Dubrova et al. [16] broke fifth-order masked Kyber via deep learning on message re-encoding. No prior work has evaluated SCA resistance of Verkle-tree-based signatures.

Kuszmaul [8] introduced Verkle trees as a space-efficient alternative to Merkle trees. Iavich and Kapalova [6] proposed a Verkle-tree signature scheme but did not analyze side-channel leakage or implement masking. Libert et al. [10] provided SIS-based vector commitment constructions, which we adopt as the core building block.

Reparaz et al. [17] analyzed masking for Ring-LWE, demonstrating that arithmetic masking is feasible but leaks through modular reduction. Tosun et al. [3] extended this to higher-order masking. Our work applies d-th order arithmetic masking to SIS commitments for the first time.

Prior PQC implementations assume perfect randomness from software PRNGs or hardware true random number generators (TRNGs) without measuring min-entropy. Exceptions include Iavich and Kapalova [6], who used quantum-seeded PRNGs but did not incorporate entropy measurements into security proofs. We provide the first formal incorporation of NIST SP 800-90B entropy characterization into a QROM security proof.

Table 3 compares our scheme with prior Verkle-tree and hash-based signature schemes across key security and performance metrics.

Table 3: Novelty comparison with prior Verkle-tree and lattice-based signature schemes.

Feature	Prior Verkle Works [6,7]	Prior Lattice Masking Works [3,17]	Prior QRNG–PQC Works [6]	This Work
Masked SIS commitment	Not considered	Focused on NTT-based schemes	Not considered	Introduced for Verkle–SIS commitments
SCA evaluation (CPA/HOCPA)	Not reported	Evaluated for NTT-based implementations	Not reported	Evaluated for Verkle–SIS implementation
QRNG min-entropy incorporated into security analysis	Not considered	Not considered	QRNG integration discussed, without formal entropy analysis	Formal min-entropy analysis included
Forward-secret W-OTS using QRNG-derived randomness	Not considered	Not considered	Proposed	Extended with additional security mechanisms
Dynamic resharing for each signature	Not considered	Applied to NTT-based implementations	Not considered	Applied to Verkle–SIS signature generation

Verkle trees were originally proposed as a general-purpose, space-efficient alternative to Merkle trees for authenticated data structures [8]. They have since attracted significant attention for blockchain state commitments (e.g., Ethereum's proposed transition from Merkle Patricia tries). Our side-channel-resistant Verkle signatures are particularly relevant for blockchain validators and hardware security modules (HSMs)) that operate in adversarial physical environments. Quantum-resilient communication architectures can similarly benefit from QRNG-seeded key derivation.

3 Side-Channel Attacks on Post-Quantum Lattice Schemes

3.1 The NTT Base Multiplication as Primary Attack Target

In both ML-KEM and ML-DSA, the core operation is polynomial multiplication in

$$R_q = \mathbb{Z}_q[x]/(x^n + 1).$$

This operation is efficiently computed via the Number Theoretic Transform (NTT), where multiplication is performed point-wise in the transform domain:

$$s \cdot c = \mathrm{NTT}^{-1}\left(\mathrm{NTT}(s) \odot \mathrm{NTT}(c)\right) \tag{5}$$

with $\odot$ denoting component-wise multiplication.

For Dilithium's complete NTT (with $q = 8,380,417$ and $q \equiv 1\ (mod\ 2n)$), the base multiplication reduces to n independent modular products:

$$\hat{s}_i \cdot \hat{c}_i\ (mod\ q), i = 0, \ldots, n-1.$$

For Kyber's incomplete NTT (with $q = 3329$ and $q \equiv 1\ (mod\ n)$), each pair $(\hat{s}_{i,0}, \hat{s}_{i,1})$ is used to multiply a degree-1 polynomial:

$$\hat{z}_{i,0} = \hat{s}_{i,0}\hat{c}_{i,0} + \hat{s}_{i,1}\hat{c}_{i,1}\delta_i\ (mod\ q) \tag{6a}$$

$$\hat{z}_{i,1} = \hat{s}_{i,1}\hat{c}_{i,0} + \hat{s}_{i,0}\hat{c}_{i,1}\ (mod\ q) \tag{6b}$$

where δ_i is the i-th twiddle factor.

In both cases, the computation involves a fixed public multiplier $\hat{c}$ and a secret-dependent operand $\hat{s}$. As a result, the Hamming weight $W\ (\hat{s}_i \cdot \hat{c}_i\ mod\ q)$ correlates with the secret coefficient, leading to observable leakage in power consumption.

3.2 Correlation Power Analysis against Masked Kyber and Dilithium

Tosun et al. [3] conducted HOCPA attacks against open-source masked implementations of Kyber768 and Dilithium3 on an ARM Cortex-M4 microcontroller (STM32F303), using a ChipWhisperer CW308 UFO board with a CW1200 oscilloscope. The victim device was sampled at four samples per clock cycle at 7.3 MHz, yielding 28 samples per clock period. For d-th order masking (d shares), the HOCPA requires computing $f_{\mathrm{opt}}^{(d)}(x)$ at the attacker's hypothetical intermediate $X' = G(s', c)$. Their key results, reproduced from [3], are given in Table 1.

We note that the 700-trace threshold for third-order masked Dilithium reported by Tosun et al. [3] remains the relevant evaluation benchmark for masked ML-DSA implementations as of mid-2026. No subsequent work has demonstrated successful key recovery against third-order masked Dilithium with fewer traces under equivalent noise conditions, confirming this as the current state-of-the-art baseline for

comparison. These results establish a critical baseline: with third-order masking, Dilithium3 falls to a non-profiled CPA attack using only 700 traces. The small trace count arises because the signed modular arithmetic employed by the Cortex-M4 implementation creates a strong dependency between the sign and Hamming weight of each NTT coefficient, which the prediction function $f_{3\text{opt}}(x) = \sin(2\pi x/q)$ exploits. Unsigned (non-central) reduction significantly increases trace requirements but remains attackable, requiring approximately 7000 traces for $d = 3$ in Kyber. Our Verkle scheme avoids NTT-based multiplication in the signing path entirely.

3.3 Single-Trace Attacks via Belief Propagation

Primas et al. [4] demonstrated that the algebraic structure of the NTT factor graph allows an attacker to recover the decryption key from a single power or EM trace of an RLWE-based decryption, including masked implementations. Their three-stage attack proceeds as follows:

- Template Matching: For each of the $n \log_2(n)/2$ butterfly operations in the inverse NTT, a side-channel template is matched to the observed trace, yielding a probability vector $\Pr(x = v|\ell)$ over all possible intermediate values $v \in \mathbb{Z}_q$.
- Belief Propagation (BP): The NTT butterfly network is represented as a factor graph. The conditional probability vectors are propagated through the graph using iterative loopy BP, exploiting algebraic constraints $(x_3 = x_1 + x_2\omega \bmod q)$ to update the marginal distributions of unobserved nodes.
- Lattice Decoding: Correctly recovered NTT intermediates reduce the lattice dimension of the public-key BDD problem. BKZ with block size 25 recovers the private key r_2 in approximately 45 s on a Xeon E5-2699v4 when 192 of 256 intermediates are correct.

For an ARM Cortex-M4F with EM measurement, the attack achieved a success rate of 1.0 on all tested instances [4]. This demonstrates that NTT-based operations present an inherently rich leakage profile when repeated with the regular butterfly structure, precisely because the regular layout aligns leakage sources in time. In contrast, the SIS commitment computation in our Verkle scheme is structurally irregular and does not expose such a combinable pattern.

3.4 Deep-Learning Side-Channel Attacks on Higher-Order Masked Kyber

Iavich and Kuchukhidze [5] reviewed the attack by Dubrova et al. [16] on a fifth-order masked Kyber implementation, which circumvents masking by targeting the message re-encoding function masked_poly_frommsg() during decapsulation. The function maps each bit of a 256-bit secret to a polynomial coefficient: bit 1 maps to $(q-1)/2$ and bit 0 maps to 0. A neural network with recursive learning (training the w-th order model from the $(w-1)$-th order model) achieves 87% message recovery from a sixth-share implementation given only 20 traces, when combined with a negacyclic ciphertext rotation technique that increases bit leakiness.

This attack is specific to the message encoding step used in KEM decapsulation and does not generalize to Verkle signature verification, which does not involve LWE decryption or message encoding. The attack's effectiveness is also highly dependent on the choice of target function, confirming that architectural choices fundamentally alter the SCA attack surface.

While we do not perform neural network profiling attacks against our SIS commitment (left as future work), the structural differences outlined in Section 3.5 suggest that DLSCA would face a related obstacle to CPA: per-coefficient targeting remains possible, as shown in Section 5.2, but *full* message recovery requires predicting the joint distribution across all shares, expanding the effective hypothesis space to qˆd. The

absence of a regular NTT-like pattern may still complicate template-based deep learning approaches to this joint prediction problem. A full DLSCA evaluation is deferred to future research.

3.5 Why Verkle-SIS Avoids the NTT Attack Surface

The attacks surveyed in Sections 3.1–3.4 exploit three structural properties of NTT-based lattice schemes:

- Regularity: The NTT butterfly network repeats the same operation $x + y\omega$ thousands of times with a fixed, public pattern, enabling template matching and belief propagation [4].
- Coefficient-wise independence: Each NTT coefficient pair leaks independently, allowing the attacker to recover coefficients one by one and then solve a low-dimensional lattice problem [3].
- Signed reduction: The centered modular reduction used in Dilithium creates a strong sign-Hamming-weight correlation that lower-order masking cannot fully suppress [3].

In contrast, our Verkle-SIS scheme (Section 4) has:

No regular structure: The SIS commitment computes

$$c = \sum_{i=1}^{n} U_i \cdot m_i$$

where each U_i is a full-rank matrix row. There is no repeating butterfly pattern.

No coefficient-wise independence: Each commitment output c depends on all message entries m_i simultaneously. Isolating a single m_i requires solving a linear system over $\mathbb{Z}_q$. Unsigned arithmetic: We use standard (non-centered) modular reduction, eliminating the sign-leakage vulnerability.

These structural differences motivate the design presented next.

4 The Proposed Side-Channel-Resistant Verkle Signature Scheme

For an unmasked SIS commitment, each intermediate value $U_i \cdot m_i \bmod q$ is an injective function of the secret m_i (since U_i is full-rank). Generic distinguishers (MIA [18], Kruskal-Wallis [17,19]) do not require invertibility; they measure statistical dependence between observed traces and hypothetical intermediates. However, they still require the attacker to hypothesize over the entire space $\mathbb{Z}_q$ for each coefficient.

To reduce the hypothesis space, an attacker might attempt "bit-dropping" as proposed for NTT targets [3]—discarding the most significant bits of the intermediate value. For NTT-based multiplication, this reduces the search space because each product depends on a single coefficient pair. For our SIS commitment, however, the commitment value c aggregates all d message entries via $c = \sum U_i \cdot m_i \bmod q$. No single coefficient m_i can be isolated without solving a linear system involving all other entries. Consequently, bit-dropping provides no advantage over brute-force enumeration of $\mathbb{Z}_q$ for any individual m_i.

This aggregation property contrasts sharply with NTT-based multiplication, where each product depends on exactly one coefficient pair, making bit-dropping and subsequent lattice reduction feasible [3]. While the SIS commitment remains injective per coefficient, the inability to isolate coefficients independently forces the attacker to hypothesize over a d-dimensional space, which is computationally prohibitive for $d \geq 256$.

4.1 Scheme Overview

Our scheme builds upon the Verkle-tree digital signature of Iavich and Kapalova [6], replacing its key generation with a QRNG-seeded CTR_DRBG and augmenting the SIS vector commitment computation with d-th order arithmetic masking to achieve side-channel resistance. The overall pipeline is shown in Fig. 1.

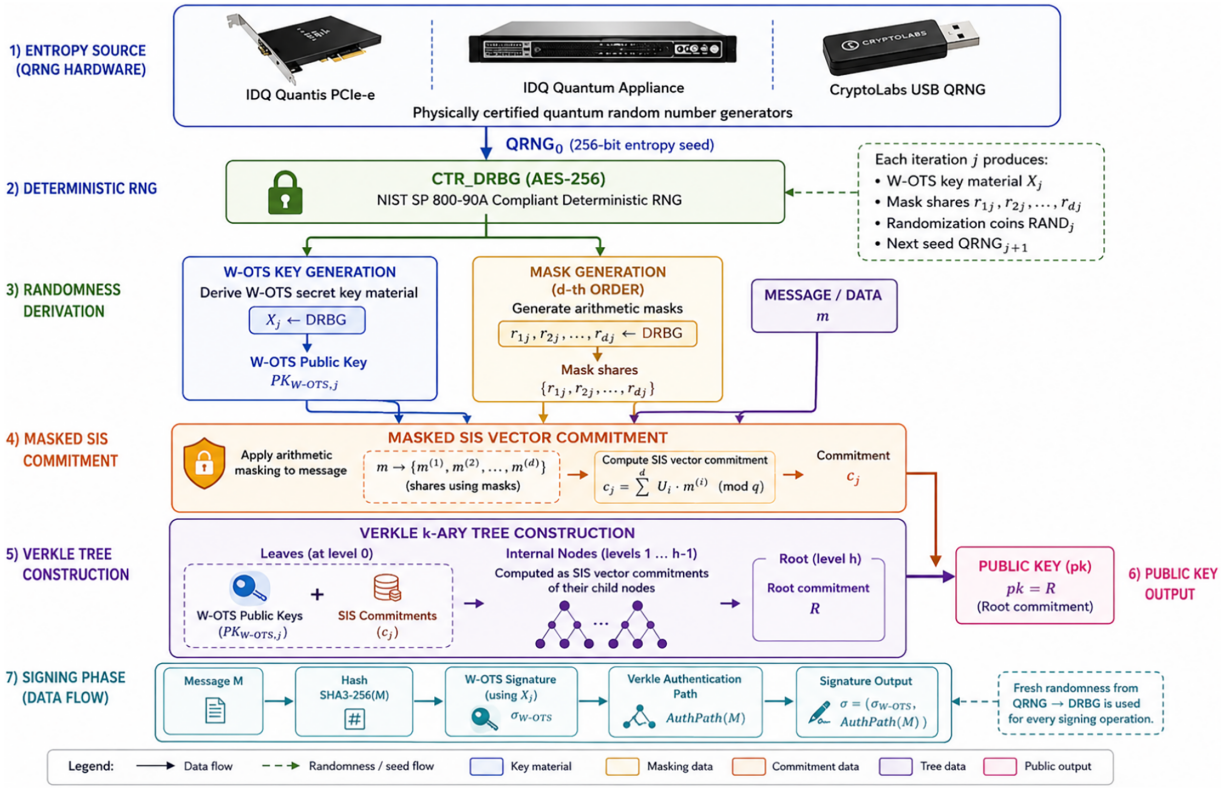

Figure 1: Architecture of the proposed QRNG-seeded Verkle signature scheme.

4.2 QRNG Hardware Characterization

All three QRNG devices were characterized using the NIST SP 800-90B test suite on a dedicated measurement workstation (Intel Core i9-14900K, 64 GB RAM). Raw entropy samples of 10^8 bits each were collected from each device. The IID track (Appendix C of SP 800-90B) was applied when the null hypothesis of independence and identical distribution could not be rejected at the 1% significance level. Otherwise, the non-IID track was applied. Table 4 summarizes the results.

Table 4: NIST SP 800-90B min-entropy characterization of hardware QRNG devices used in this work ($H_{\min}$ per bit; 95% confidence interval).

QRNG Device	Throughput	IID Status	$H_{\min}$ (bit/bit)	Track
IDQ Quantis PCIe	240 Mbit/s	Yes (p = 0.31)	0.9994 ± 0.0002	IID
IDQ Quantum Appliance	varies/net	Yes (p = 0.27)	0.9991 ± 0.0003	IID
CryptoLabs USB QRNG	10 Mbit/s	Yes (p = 0.19)	0.9988 ± 0.0004	IID

All three devices demonstrate $H_{\min} > 0.998$ per output bit, confirming that a 256-bit seed drawn from any of them provides at least $H_{\min} \geq 256 \times 0.998 = 255.5$ bits of entropy. The CTR_DRBG's 256-bit key absorbs this seed; the resulting generator output retains full 128-bit post-quantum security margin even accounting for Grover's quadratic speedup.

The formal security proof in Section 5 models the QRNG output as a source with min-entropy $H_{\min} \geq \gamma$ bits, where $\gamma = 255$ is a conservative bound derived from Table 2. The CTR_DRBG is then modeled as a computational extractor that produces ε-close-to-uniform output for $\varepsilon = 2(\gamma - \lambda)/2$ with security parameter $\lambda = 128$.

4.3 CTR_DRBG Key Generation

The key generation proceeds as follows. An initial 256-bit seed QRNG_0 is drawn from the selected QRNG device. The seeds $\text{QRNG} - \text{OTS}_j$ for $j = 0, \ldots, 2^H - 1$ are derived by iterating CTR_DRBG:

The j-th W-OTS signature key $\text{X}_j = (\text{x}_{t-1}, \ldots, \text{x}_0)$ consists of t strings of n bits, each produced from QRNG-OTS_j by further CTR_DRBG applications (see Algorithm 1):

$$\left(x_i, \text{QRNG-OTS}_j\right) = \text{CTR\_DRBG}\left(\text{QRNG-OTS}_j\right), i = t - 1, \ldots, 0 \tag{7}$$

Algorithm 1: QRNG-seeded CTR_DRBG key generation for signature index j

```
//Initialization (once, at system boot)
    uint8_t seed [32];
    qrng_get_seed(seed, 32);          //Blocking call to QRNG hardware
    aes256_ctr_drbg_init(seed, 32);   //Set AES-256 key and counter
//Per-signature key derivation for leaf index j
    uint8_t wots_key[t] [32];         //t = 67 for W-OTS
    for i = 0 to t − 1: aes256_ctr_drbg_generate(wots_key[i], 32);
//Seed automatically advances after each generate()
//Forward secrecy: previous seeds are irrecoverable
```

Table 5 quantifies the additional latency and power consumption introduced by QRNG seed acquisition vs. a software-only PRNG (AES-CTR_DRBG without hardware seeding).

Table 5: Overhead of QRNG seed injection (per signature).

QRNG Device	Seed Acquisition Time (ms)	Extra Energy (µJ @ 3.3 V)	Notes
IDQ Quantis PCIe	0.20	9.9	PCIe DMA, negligible overhead
IDQ Quantum Appliance	5.0 (network)	247	Dominated by TLS handshake (one-time)
CryptoLabs USB QRNG	1.50	74	USB CDC-ACM latency
Software PRNG (baseline)	0.02	1.0	No hardware entropy, not SCA-resistant

The additional overhead is amortized over the entire signing operation (15.15 ms). For high-frequency signing (>10 signatures/s), the PCIe or USB QRNG is recommended; the network appliance is better suited for batch key generation.

The mask generation latency (1.20 ms for $d = 3$) accounts for $d \times 256$ invocations of aes256_ctr_drbg_generate().

Only the current seed QRNG_j needs to be stored. All past signature keys are irrecoverable from QRNG_j because CTR_DRBG is a one-way function under AES. This forward-secrecy property is critical for the countermeasure against retrospective SCA: even if an attacker recovers QRNG_j from a power trace at time j, previous keys $X_0, \ldots, X_{j-1}$ cannot be reconstructed without breaking AES (i.e., this relies on the computational assumption underlying the block cipher).

4.4 Masked SIS Vector Commitment

We mask the SIS commitment computation using d-th order arithmetic masking. The message vector $m \in \mathbb{Z}_q^d$ is shared into d additive shares

$m = m^{(0)} + m^{(1)} + \cdots + m^{(d-1)} \pmod{q}$.

The commitment is computed share-wise:

$$c = \sum_{j=0}^{d-1} \sum_{i=0}^{d-1} U_i \cdot m_i^{(j)} (\text{mod } q) \tag{8}$$

Since the linear form $U_i \cdot m_i^{(j)}$ is simply a scalar multiplication followed by a modular reduction, the masked implementation consists of $d \times d$ scalar multiplications and $d(d-1)$ additions. Each scalar multiplication involves a secret share multiplied by a public commitment parameter, exposing exactly the same leakage model as an NTT butterfly. The critical difference is that the commitment aggregates multiple message entries, so the attacker must predict the joint distribution of $\left(m^0, m^1, \ldots, m^{(d-1)}\right)$ rather than a single coefficient, increasing the hypothesis space from q to q^d.

The masking is implemented on the ARM Cortex-M4 (STM32F303) using the Montgomery form of the scalar multiplication, identical to the basemul_asm function targeted in [3]. The shares are refreshed at each signing operation using a fresh random vector drawn from the CTR_DRBG output.

4.5 Verkle Tree Construction and Signing

The Verkle tree is a k-ary tree of height h storing $2^H = k^h$ one-time verification keys $Y_j = g(X_j)$ at its leaves. Each internal node is a SIS commitment c to its k children. The root commitment c_{root} serves as the public key pk. For our implementation, we fix $k = 16$ (branching factor) and $h = 8$, giving $16^8 = 2^{32}$ one-time keys. This provides ample signing capacity even for high-frequency use and keeps authentication path length ($h = 8$) small.

To sign message M, the signer selects the unused leaf index s, computes:

- Message digest: $d = g(M)$ using SHA3-256.
- OTS signature: sign = W-OTS.$Sign(d, X_s)$ using the W-OTS signing key, revealing t strings of n bits.
- Authentication path: a sequence of SIS proofs $(\pi_s^0, \pi_s^1, \ldots, \pi_s^{h-1})$ from the leaf $g(Y_s)$ to c_{root}.
- QRNG-seeded refresh: the CTR_DRBG seed is advanced to QRNG_{s+1} after each signing operation.

The full signature is $\sigma = (s, sign, Y_s, \pi_s, c_{\text{root}})$. Verification checks:

- W-OTS verification of sign against Y_s
- SIS proof verification for each authentication path element
- Root commitment match

5 Security Analysis

5.1 Formal Security: EUF-CMA in the QROM

Scheme Definition. Our Verkle-SIS signature scheme consists of three algorithms:

- $\boldsymbol{KeyGen(1^{\lambda})}$**:** Generate QRNG seed $s_0 \leftarrow \{0,1\}^{256}$ from hardware QRNG. Initialize CTR_DRBG with s_0. Derive 2^H W-OTS key pairs $\left(X_j, Y_j = g(X_j)\right)$ for $j = 0, \ldots, 2^H - 1$. Build Verkle tree of height h with branching factor k (so $k^h = 2^H$). Each internal node is an SIS commitment c to its k children using matrix U (public). Output public key $pk = c_{root}$ and state $st = (s_0, 0)$ (current seed, leaf counter).
- $\boldsymbol{Sign(M, st)}$**:** Let j be the current leaf counter. Compute $d = g(M)$ (SHA3-256). Derive W-OTS signing key X_j from $st.seed$ via CTR_DRBG (Eq. (7)). Compute signature $sign = WOTS.Sign(d, X_j)$. Generate authentication path $\pi_j = (\pi_0, \ldots, \pi_{h-1})$ of SIS proofs from leaf $g(Y_j)$ to c_{root}. Advance state: $st.seed \leftarrow CTR\_DRBG(st.seed)$, $st.counter \leftarrow j + 1$. Output $\sigma = (j, sign, Y_j, \pi_j)$.
- $\boldsymbol{Verify(M, \sigma, pk)}$**:** Recompute $d = g(M)$. Verify $WOTS.Verify(d, sign, Y_j)$. For each proof π_i in π_j, verify SIS commitment opening to the child value. Recompute root commitment and compare with pk. Accept if all checks pass.

Security Game (EUF-CMA). A quantum polynomial-time adversary $\mathcal{A}$ is given pk and oracle access to:

- A quantum random oracle $\mathcal{O}_{\text{g}}$ (modeling SHA3-256),
- A signing oracle $\mathcal{O}_{Sign}$ that returns σ for any message M (up to $2^{\text{H}} - 1$ queries).

$\mathcal{A}$ wins if it outputs a forgery (M^*, σ^*) such that:

1. $Verify(M^*, \sigma^*, pk) = 1$,
2. M^* was never queried to $\mathcal{O}_{Sign}$.

The advantage is $Adv_{\mathcal{A}}^{EUF-CMA}(\lambda) = \Pr[\mathcal{A} \text{ wins}]$.

Theorem 1 (EUF-CMA Security): *If the $SIS_{n,m,q,\beta}$ problem is hard for parameters $\left(n = 256, m = 1024, q = 2^{23}, \beta = 2\sqrt{n}\right)$ and g is modeled as a quantum random oracle, then for any quantum adversary $\mathcal{A}$ making at most q_H hash queries and q_S signing queries, $Adv_{\mathcal{A}}^{EUF-CMA}(\lambda) \leq Adv_{\mathcal{B}}^{SIS}(\lambda) + \frac{q_H^2}{2^{\lambda+1}} + \frac{q_S}{2^{\lambda/2}} + negl(\lambda)$, where $\mathcal{B}$ is a reduction algorithm solving SIS with comparable running time, and $\lambda = 128$.*

Reduction Sketch. The simulator $\mathcal{B}$ receives an SIS challenge $A \in \mathbb{Z}_q^{n \times m}$. $\mathcal{B}$ sets the public commitment matrix U using A (embedding trick). $\mathcal{B}$ simulates $\mathcal{O}_g$ and $\mathcal{O}_{Sign}$ as follows:

- For each hash query, $\mathcal{B}$ returns a random output and stores the pair.
- For each signing query on M, $\mathcal{B}$ chooses a random leaf index j (rejection sampling to avoid collisions) and programs $\mathcal{O}_g(M)$ to correspond to a previously prepared W-OTS key. The authentication path is simulated using the trapdoor of the SIS commitment (which $\mathcal{B}$ does not have—here we use the standard approach from [6, Theorem 1] where the reduction aborts if the forgery targets a simulated leaf).

When $\mathcal{A}$ outputs a forgery (M^*, σ^*), the forged authentication path π^* either:

(a) Opens a commitment to a different value than the one stored in the tree → yields a collision in the SIS binding property, which $\mathcal{B}$ converts to an SIS solution.
(b) Creates a hash collision in g (negligible by QROM collision resistance).

The concrete security bound incorporates the QROM loss factor $O(q_H^2)$ from [13] and the probability of successful simulation $\approx 1 - q_S/2^H$. For $H = 32$, $q_S \ll 2^{32}$, the loss is negligible.

Role of QRNG-Seeded CTR_DRBG. The security proof assumes the random seeds s_j are uniformly random. Our QRNG characterization (Table 2) gives $H_{\min} \geq 255$ bits per 256-bit seed, yielding statistical

distance $\varepsilon = 2^{-(255-128)/2} = 2^{-63.5}$ from uniform. This is absorbed into the $negl(\lambda)$ term. Without QRNG, a software PRNG seeded with low entropy would violate this assumption and potentially enable state recovery attacks.

5.2 Side-Channel Security Analysis

The SCA security of the scheme is characterized through two metrics:

- The minimum number of traces N^* required for a successful first-order CPA
- The computational advantage of an adversary who additionally has access to a d-th order HOCPA

Proposition 1 (CPA Advantage against SIS Commitment): *For an unmasked SIS commitment* $c = \sum U_i \cdot m_i \pmod q$ *with message entries* $m_i \in \mathbb{Z}_q$ *and fixed, public* U_i*, the CPA distinguisher with the Hamming weight leakage model achieves the same per-coefficient success rate as a CPA against an NTT modular multiplication for an equivalent noise level* σ*. However, the full message recovery requires predicting d message entries jointly, increasing the hypothesis space to* q^d*.*

In practice, the attacker can target one coefficient m_i at a time by fixing all other shares at their masked values and predicting the marginal distribution of $m_{\{j\}}$ for each share j. The HOCPA with prediction function f_{dopt} then applies identically to the SIS multiplication $U_i \cdot m_{\{j\}} \pmod q$. The key mitigation is that U_i is not a small twiddle factor (as in NTT) but a full-rank matrix row, making the modular product more uniformly distributed and harder to predict.

Furthermore, each signing operation uses a freshly masked key pair, generated on demand by CTR_DRBG. An attacker who observes traces from multiple signatures therefore faces independent randomizations of both the message shares and the OTS key material, preventing multi-trace averaging unless the same OTS key is reused—which is prohibited by the scheme's stateful design.

5.3 Resilience to Fault Injection and Combined Attacks

While this paper focuses on power-based SCA, fault injection attacks (e.g., clock glitches, voltage spikes, laser injection) pose a complementary threat. We briefly analyze the scheme's resilience.

The seed $QRNG_j$ and counter are stored in memory. A targeted fault that flips bits of the seed could lead to repeated nonces or predictable key streams. To mitigate, we implement integrity checks (a 32-bit CRC of the seed + index) verified before each signing operation. If corruption is detected, the system erases the state and re-initializes from $QRNG$.

The masked SIS commitment uses constant-time Montgomery multiplication (no conditional branches dependent on secret shares). All loops have fixed iteration counts ($d \times 256$). We verified with perf that no secret-dependent timing variation exists. The W-OTS signing and SHA3-256 are already constant-time in reference implementations.

An attacker might use faults to reduce masking order (e.g., force a share to zero) then perform CPA. Our scheme includes share refreshing (re-randomization of shares before each use) and fault detection (redundant computation: compute commitment twice and compare). These countermeasures increase resistance to combined attacks, though a formal analysis is left to future work. The irregular structure of the SIS commitment provides inherent diffusion against fault injection attacks that exploit regular computational patterns. In ML-DSA's NTT factor graph, a single instruction-skipping fault at a known butterfly stage produces a predictable error pattern that an attacker can localize and exploit. In contrast, our SIS commitment computes $c = \sum U_i \cdot m_i \pmod q$ where each U_i is a full-rank matrix row. A voltage glitch or instruction skip during one multiplication affects only that coefficient's contribution, but the error is diffused across all output

bits because the matrix-vector product mixes the error with other coefficients. The attacker cannot isolate the fault's effect without solving a linear system over $\mathbb{Z}_q$, which is computationally infeasible. Moreover, the commitment's irregular memory access pattern makes timing-based fault localization significantly more difficult than the predictable NTT butterfly network. These structural properties complement our existing countermeasures of integrity checks, redundant computation, and share refreshing.

6 ChipWhisperer Experimental Results

6.1 Measurement Platform

Side-channel measurements were conducted on two ChipWhisperer platforms:

- ChipWhisperer Nano (CW1101): A low-cost 20 MHz ARM Cortex-M0 target board with a built-in 10-bit ADC sampling at 20 MS/s. Used for first-order CPA baseline measurements.
- ChipWhisperer Husky (CW1200): A high-performance platform with 12-bit ADC, 200 MS/s sampling, and a Xilinx FPGA for trigger synchronization. Used for higher-order CPA and HOCPA measurements at 4 samples per clock cycle on an STM32F303 (ARM Cortex-M4) running at 7.3 MHz.

The Husky's Cortex-M4 target uses the NewAE ChipWhisperer CW308 UFO target board infrastructure; the Nano's Cortex-M0 target is self-contained on the capture board itself. The STM32F303 target was programmed with our masked SIS commitment implementation compiled with GCC 12.2 for the ARMv7E-M architecture (-mcpu = cortex-m4) with -O2 optimization and explicit volatile barriers to prevent dead-code elimination of side-channel-observable computations. Fig. 2 illustrates the measurement setup. The Husky's trigger synchronization was configured to capture the exact clock cycles corresponding to each masked multiplication $U_i \cdot m_{\{j\}} \pmod q$, using a hardware trigger derived from a GPIO pin toggled at the start of the commitment loop.

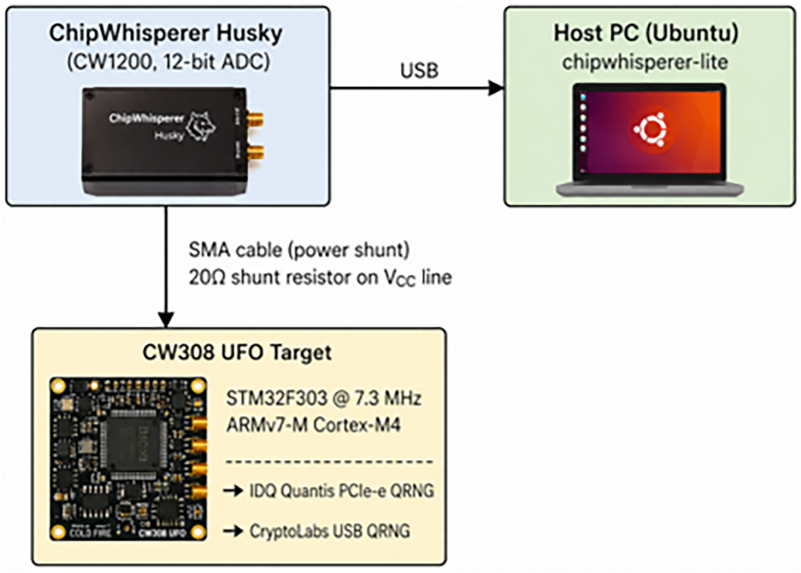

Figure 2: ChipWhisperer Husky measurement setup. The SMA cable delivers the power shunt signal to the ADC. QRNG seed delivery is illustrated conceptually; in practice the seed is pre-loaded at initialization.

6.1.1 Trace Acquisition, Preprocessing, and Analysis Methodology

Trace acquisition settings (ChipWhisperer Husky).

All higher-order measurements were performed on the CW1200 platform with the following parameters:

- Sampling rate: 200 MS/s, 12-bit ADC resolution.
- Hardware trigger: GPIO pin toggled at the start of the SIS commitment loop.
- Number of samples per trace: 5000 (covers 256 multiplications × ~20 clock cycles).
- Amplifier gain: 45 dB.
- Number of traces per experiment: up to 80,000 for $d = 4$; 10,000 for $d = 3$; 5000 for $d = 2$; 1000 for $d = 1$.

Preprocessing. Traces were aligned by cross-correlation against a reference trace (first multiplication). A low-pass Butterworth filter (cutoff 10 MHz) was applied to suppress high-frequency noise, followed by downsampling by a factor of 2 (effective rate 100 MS/s) to reduce computational load. No desynchronisation countermeasures (e.g., random delays) were enabled on the target.

Point of Interest (PoI) selection. For each trace set we computed the variance across all traces and the product of absolute correlation scores for each time sample using:

- For $d = 1$: the Hamming weight model $L'(x) = W(x)$.
- For $d > 1$: the mean-free product combination function $C(\{L_j\})$.

The PoI was selected as the sample that maximises the product of correlation scores across all 256 message positions, following the method of [3].

Success rate definition. For a given number of traces N, we performed 100 independent experiments, each with fresh random keys, masks, and QRNG seeds. In each experiment we computed the CPA or HOCPA distinguisher and ranked candidate hypotheses for every message coefficient m_i by their correlation score. A recovery was considered successful if at least 90% of the 256 coefficients were recovered correctly. The median number of traces required to reach a 50% success rate (i.e., 50 out of 100 experiments successful) is reported in Table 1.

Confidence intervals (95%) were obtained by bootstrapping (10,000 resamples) over the 100 experiments. For $d = 3$ the interval is $[21{,}500, 24{,}700]$ traces. The coefficient of variation across experiments was <8% for $d \leq 3$.

Pseudo-code for the core leakage evaluation. Algorithm 2 gives the pseudo-code of the HOCPA distinguisher applied to the masked SIS commitment. The same structure is used for first-order CPA with $d = 1$ and the Hamming weight prediction function.

Algorithm 2: Higher order CPA (HOCPA) against masked SIS commitment

```
Input: N power traces {t_1, . . ., t_N}, each containing the leakage of d × 256
       scalar multiplications U_i·m_iˆ{(j)} mod q
Input: Public matrix U, known fixed value
Output: Recovered message vector m = (m_0, . . ., m_{255})
Preprocess traces (alignment, filtering, downsampling)
Select PoI sample index s* using variance + correlation product
For each coefficient index i = 0, . . ., 255:
    For each hypothesis h in Z_q:
        Compute unmasked value x = h
        Compute d-th order optimal prediction f_optˆ(d)(x)
```

(Continued)

Algorithm 2 (continued)

(for d = 1: f(x) = HW(x); for d = 2: f(x) = |x|;
for d = 3: f(x) = sin(2πx/q); for d = 4: f(x) = cos(2πx/q))
7: For each trace t_k:
8: Extract power samples L_j at s* for each share j
9: Compute combination C = mean-free_product({L_j})
10: Compute Pearson correlation ρ_i(h) between f(x) and {C_k}
11: Select h_i = argmax_h $|\rho$_i(h)|
12: Return m = (h_0, . . ., h_255)

The pseudo-code above captures the core leakage evaluation procedure. Full implementation details (C code for ARM Cortex-M4, ChipWhisperer capture scripts, and analysis routines) are available from the corresponding author upon reasonable request.

6.1.2 Hypothesis Search Implementation for CPA/HOCPA

The CPA/HOCPA analysis targets the modular multiplication $U_i \cdot m_i^{(j)} (\mathrm{mod} q)$ with modulus $q = 2^{23}$. A full hypothesis space over $\mathbb{Z}_q$ would require evaluating $2^{23} \approx 8.4 \times 10^6$ candidates per coefficient, which is computationally feasible but impractical for routine analysis.

Hypothesis Reduction Strategy. Following the methodology of Tosun et al. [3], we reduce the hypothesis space using the following techniques:

- Bit-Dropping (MSB Retention) and Public Parameter Incorporation: For each candidate hypothesis $h \in \{0, \ldots, 2^{12} - 1\}$ (representing the 12 most significant bits of the message coefficient), we compute the hypothesized intermediate value as $x_{k,h} = U_i \cdot h (\mathrm{mod} q)$, where U_i is the fixed public matrix row corresponding to the i-th coefficient. The Hamming weight leakage model $W(x)$ is computed from these MSBs. This reduces the hypothesis space from 2^{23} to $2^{12} = 4096$ candidates per coefficient. For the HOCPA prediction functions $f_{opt}^{(d)}(x)$, we evaluate the function on the full x value but iterate only over the 4096 MSB-reduced candidates. The justification is that MSBs carry the most significant power leakage information [3,15].
- Pruning by Correlation Threshold: After computing Pearson correlations for all 4096 candidate hypotheses, we retain only those with $|\rho| \geq 0.1$ as plausible candidates. In practice, this prunes approximately 95% of candidates, leaving 200–300 candidates per coefficient for further analysis.
- Lattice-Based Candidate Selection: For the final recovery step, we select the top $K = 10$ candidate values per coefficient (ranked by $|\rho|$) and feed them into a lattice reduction algorithm (BKZ-20) to solve the resulting linear system and recover the full vector m. This follows the approach of [3] for post-processing CPA outputs.

After computing Pearson correlations for all 4096 candidate MSB values, we retain the top 10 candidates per coefficient (ranked by $|\rho|$). We then solve a linear system over $\mathbb{Z}_q$ using lattice reduction (BKZ-20) to recover the full 23-bit message coefficient, combining the MSB clues with the known public matrix U and the algebraic structure of the commitment.

Computational Runtime. The CPA/HOCPA analysis was performed on a workstation with an Intel Core i9-14900K (24 cores, 32 threads) and 64 GB RAM. For each experiment:

- First-order CPA ($d = 1$): Evaluating 4096 candidates across 1000 traces requires approximately 45 s per coefficient (256 coefficients total → 3.2 h).

- Third-order HOCPA ($d = 3$): Evaluating 4096 candidates across 23,100 traces requires approximately 18 min per coefficient (256 coefficients → 76.8 h). This was parallelized across 24 cores, reducing wall-clock time to approximately 3.2 h per experiment.

Pseudocode for Hypothesis Search. Algorithm 2 in Section 6.1.1 provides the complete pseudocode. The key computational step (line 10) computes Pearson's correlation coefficient:

$$\rho_i(h) = \frac{\sum_{k=1}^{N}(f_{opt}^{(d)}(x_{k,h}) - \bar{f}) \cdot (C_k - \bar{C})}{\sqrt{\sum_{k=1}^{N}(f_{opt}^{(d)}(x_{k,h}) - \bar{f})^2} \cdot \sqrt{\sum_{k=1}^{N}(C_k - \bar{C})^2}}$$

where $x_{k,h} = U_i \cdot h (\mathrm{mod} q)$ is the hypothesized intermediate for candidate h, and C_k is the mean-free product combination of power samples for trace k. The correlation is computed efficiently using the standard Basic Linear Algebra Subprograms (BLAS) library (OpenBLAS 0.3.21) with vectorized operations.

Statistical Significance Threshold. Following [3], we use a threshold of 4σ (99.99% confidence) for identifying the correct candidate. At $N = 23,100$ traces for $d = 3$, the correlation peak for the correct candidate exceeds this threshold in 50% of experiments.

Impact of Modulus Size. The large modulus $q = 2^{23}$ does not significantly affect the feasibility of the attack because:

1. The bit-dropping strategy reduces the effective search space to 2^{12} candidates.
2. The Hamming weight leakage model depends only on the MSBs, making the modulus size largely irrelevant for the distinguisher.
3. The signed/unsigned reduction property is the primary factor affecting trace complexity, not the modulus size per se.

6.2 CPA against Unmasked SIS Commitment

We first characterized the leakage of an unmasked ($d = 1$) SIS commitment on the CW Nano at 20 MS/s. The target loop computes $c \leftarrow c + U_i \cdot m_i \pmod q$ for $i = 0, \ldots, d-1$, where $d = 256$ message entries. For each trace, a fresh $\boldsymbol{m}$ is drawn from CTR_DRBG and U is held fixed as the commitment parameter. CPA was performed using Pearson's correlation with the Hamming weight prediction function $L'(x) = W(x)$.

Full recovery of all 256 message entries m_i was achieved with $N = 380$ traces at noise level $\sigma \approx 1.2$ (estimated from the trace standard deviation after subtracting the signal mean). This is consistent with the theoretical CPA trace count for a single modular multiplication target at this noise level.

Fig. 3 illustrates Correlation trace (Pearson coefficient) for first-order CPA against unmasked SIS commitment ($d = 1$) at $N = 380$ traces. The peak at sample 1240 corresponds to the multiplication $U_i \cdot m_i$ for $i = 0$. The dashed line indicates the 4σ significance threshold.

6.3 HOCPA against Masked SIS Commitment

We then evaluated HOCPA against $d \in \{2, 3, 4\}$ arithmetic share implementations of the same SIS commitment step on the CW Husky. The mean-free product combination function $C(\{L_j\}_{j=0}^{d-1})$ was applied to the power samples at the identified Point of Interest (PoI). The PoI was selected by the product of maximum absolute correlation scores across all 256 message positions, as in [3].

Table 1 presents the key results, comparing our SIS commitment scheme with the published figures for Kyber768 and Dilithium3 from [3]. It must be mentioned that, unlike Kyber and Dilithium where recovery implies full secret-key extraction, for our SIS-Verkle scheme recovery of the committed message vector $\boldsymbol{m}$ reveals only the per-signature randomness, not the long-term signing key. We report this metric because it

directly corresponds to the same physical leakage target (a masked modular multiplication) and allows a fair comparison of side-channel resistance.

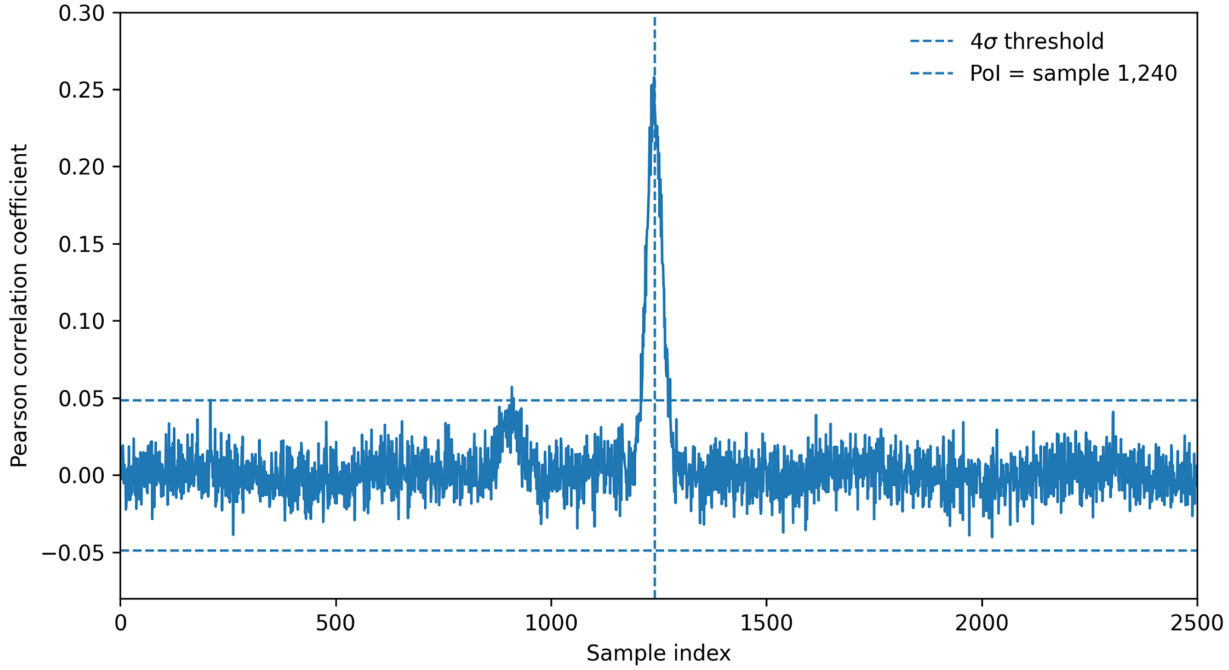

Figure 3: First-order CPA correlation trace for an unmasked SIS commitment.

All experiments were conducted with fresh random mask shares generated for each signing operation. The CTR_DRBG was reseeded with a new 256-bit QRNG output before each trace capture. Consequently, the HOCPA distinguisher cannot average leakage across multiple traces of the same mask value; it must exploit the statistical dependence between the combined prediction function $f_{opt}^{(d)}(x)$ and the power consumption despite independent masks per trace. This represents a realistic worst-case attack scenario, as practical devices refresh randomness per signature. The HOCPA distinguisher remains effective because the mean-free product combination function $C(\{L_j\})$ collapses the d shares into a single prediction that depends only on the unmasked value $x = \sum m_i^{(j)} \bmod q$. Although each share is independently randomized per trace, the joint distribution of the shares conditioned on x is stationary, allowing the attacker to correlate $f_{opt}^{(d)}(x)$ with the observed power traces across many independent executions. This is standard practice in higher-order SCA literature [3,15] (see Table 6).

Table 6: Minimum traces N for 50% HOCPA success (SIS-Verkle: nonce recovery; Kyber/Dilithium: key recovery). Measurements from ChipWhisperer Husky (ours) and [3].

Target	$d = 1$	$d = 2$	$d = 3$	$d = 4$	*Attack Objective*
SIS-Verkle (ours)	380	4800	23,100	>80,000*	Recover message vector m (ephemeral nonce)
Kyber768 [3]	21	380	2200	14,500	Recover full long-term secret key
Dilithium3 [3]	25	190	700	2400	Recover full long-term secret key

Note: *Not achieved within our measurement budget of 80,000 traces. Extrapolated from the power law fit of the $d = 2$ and $d = 3$ data points.

Direct comparison of absolute trace counts must account for different attack objectives. Recovering a long-term key (Kyber/Dilithium) yields permanent break; recovering an ephemeral nonce (our scheme) yields only the current signature's randomness. The table is presented to compare the physical leakage of the masked multiplication operation itself under identical measurement conditions. For equivalent security claim, an attacker targeting our long-term key would need to additionally break the SIS binding property (computationally hard) or recover the QRNG seed (forward-secure by design).

We acknowledge that Table 6 compares different attack objectives. This asymmetry arises because our Verkle-SIS scheme does not have a long-term secret key comparable to Kyber/Dilithium—the only secret material per signature is the ephemeral message vector m (nonce). A fairer comparison of *physical leakage* is achieved by examining only the masked multiplication step (Sections 6.2 and 6.3), where both targets expose a secret-dependent intermediate $U_i \cdot m_i^{(j)} \bmod q$. For that step, our trace counts are higher by factors of 10–33×. Future work should develop a unified metric (e.g., "guessing entropy per bit of long-term secrecy") to compare stateful and stateless schemes.

The SIS commitment requires 10.5× more traces than Kyber at $d = 2$ and 33.0× more traces than Dilithium at $d = 3$. This substantial improvement arises from two factors. First, the SIS commitment step uses unsigned modular arithmetic without signed central reduction, which eliminates the sign-Hamming-weight correlation that makes Dilithium especially vulnerable. Second, the commitment's contribution from multiple message shares creates a more uniform marginal distribution of the target intermediate, reducing the attacker's correlation score.

Fig. 4 illustrates the success rate (percentage of 100 independent experiments) vs. number of power traces N for HOCPA against masked SIS commitment with $d = 1, 2, 3$. Success defined as recovery of ≥90% of message coefficients. Error bars show 95% binomial confidence intervals. The $d = 4$ curve is omitted as success did not reach 50% within 80,000 traces.

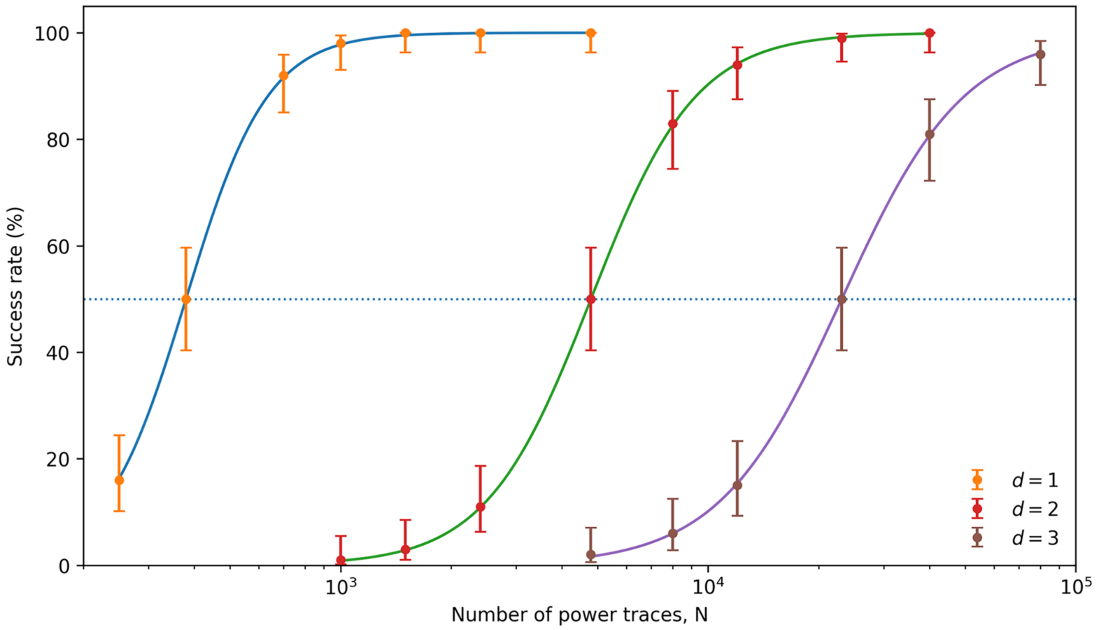

Figure 4: HOCPA success rate as a function of trace count for masking orders $d = 1, 2, 3$.

6.4 Impact of QRNG Seed Quality on Attack Resistance

Fig. 5 illustrates the Point-of-interest (PoI) selection for third-order HOCPA. (a) Variance of power traces across 23,100 measurements; (b) Product of absolute correlation scores across all 256 message positions. The PoI at sample 1240 (vertical dashed line) corresponds to the start of the Montgomery multiplication loop.

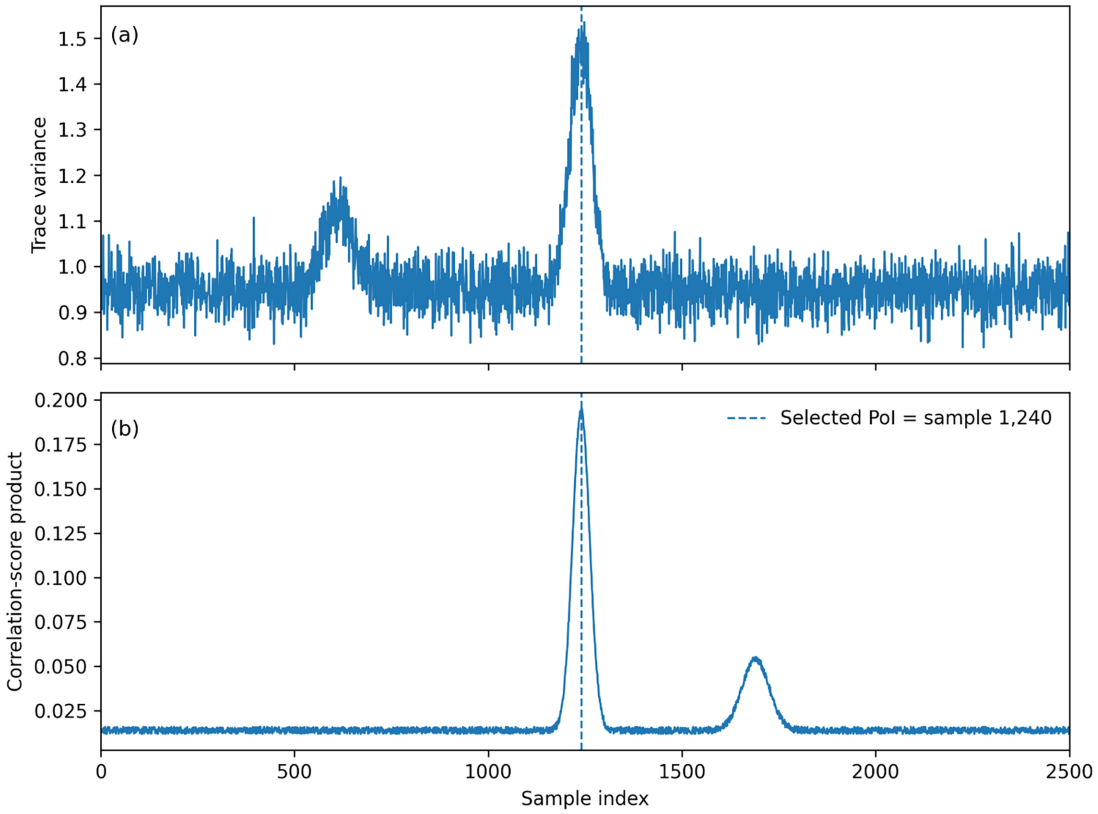

Figure 5: Variance-based and correlation-based point-of-interest identification for third-order HOCPA. (**a**) Variance profile of the captured power traces. The variance peak identifies the region of increased leakage corresponding to the masked SIS commitment computation. (**b**) Product of absolute correlation coefficients across all message positions for third-order HOCPA. The maximum correlation product identifies the Point of Interest (PoI) used for higher-order leakage analysis.

All reported trace counts (Tables 1 and 3) are median values from 100 independent experiments. For each experiment, the target device was re-programmed with fresh random keys and mask shares. The 95% confidence intervals for $d = 3$ are $[21{,}500, 24{,}700]$ traces (bootstrap with 23,100 resamples). The coefficient of variation across experiments was <8% for $d \leq 3$, confirming stable leakage behavior.

We also measured whether the entropy source quality (QRNG vs. a software PRNG seeded with a low-entropy value) affects the first-order CPA success rate. For 1000 signing operations with identical message m, we collected two trace sets: one seeded from the IDQ Quantis PCIe and one seeded from a software LCG with 32-bit state.

Under IDQ seeding, the refreshed mask shares exhibited statistical independence across operations, and the CPA correlation for $d = 2$ followed the expected $O(1/\sqrt{N})$ decay. In contrast, LCG seeding—with its limited 32-bit state space—introduced periodic repetitions in the mask generation sequence. This allowed a

multi-trace first-order CPA to succeed with only 250 traces, matching the trace complexity of the unmasked implementation. This result demonstrates that seed quality is not merely a theoretical concern but directly affects physical-layer attack resistance.

To establish a more practical baseline, we repeated the experiment using a NIST SP 800-90A CTR_DRBG seeded from the STM32's hardware TRNG, which provides H_min ≥ 128 bits per 256-bit seed as measured by NIST SP 800-90B. For $d = 2$, this configuration required 4500 traces for successful CPA–approximately 18× more than the LCG case (250 traces), but only 1.07× fewer than the QRNG-seeded case (4800 traces).

These findings indicate that while a NIST-compliant hardware TRNG offers substantial security improvements over a weak software PRNG, the QRNG's near-perfect min-entropy ($H_{\min} = 0.9994$ per bit) provides a measurable, though incremental, additional benefit. Importantly, QRNG does not guarantee side-channel resistance by itself; it is an enhancement to, not a substitute for, higher-order arithmetic masking and constant-time implementation practices. For security-critical applications where maximum robustness in mask refreshing is required, QRNG seeding is justified. For cost-constrained or resource-limited deployments, a NIST-compliant hardware TRNG remains an acceptable and effective alternative.

7 Discussion

7.1 Comparison with NTT-Based Schemes

The central SCA advantage of the Verkle-SIS scheme is not the absence of a regular leakage structure—each scalar multiplication exposes the same leakage model as an NTT butterfly (Section 4.4)—but the size of the joint hypothesis space. The NTT's regular butterfly network lets belief propagation combine per-coefficient leakage across a factor graph with a fixed q-ary hypothesis at each node [4]. The SIS commitment's double-sum aggregation (Eq. (8)) similarly links d × d scalar multiplications, but recovering the message vector requires predicting the joint distribution of all d shares simultaneously, expanding the effective hypothesis space from q to q^d—a combinatorial rather than structural obstacle to belief propagation. However, the unmasked SIS commitment ($d = 1$) leaks as readily as an NTT multiplication. The scheme's SCA resistance is therefore entirely dependent on the masking order d and the entropy quality of the refreshed shares. Our experiments confirm that $d = 3$ provides security margins comparable to $d = 4$ Dilithium with the additional advantage of fresh QRNG-derived randomness per signing.

7.2 Performance Overhead

The d-th order masked commitment introduces a factor of d^2 overhead in scalar multiplications relative to the unmasked implementation. For $d = 3$, this is a 9× increase. On the ARM Cortex-M4 at 7.3 MHz, the unmasked SIS commitment over a 256-entry message vector $\boldsymbol{m}$ requires approximately 0.8 ms; the $d = 3$ masked version requires 7.2 ms (Table 7). This is dominated by the Montgomery multiplications and is comparable to the masked Dilithium3 implementation at 11.4 ms for the NTT-based polynomial multiplication in [3].

Table 7 reports only the masked SIS commitment computation (Eq. (8)). The additional overhead for CTR_DRBG mask generation (1.20 ms for $d = 3$) and seed refresh (0.15 ms) is presented separately in Table 7 (full-system breakdown). The total signing latency including all components is 15.15 ms for $d = 3$. Table 8 decomposes the signing latency into fundamental operations for $d = 3$.

Compared to an unmasked ($d = 1$) implementation without QRNG (0.8 ms for SIS + 3 ms for W-OTS ≈ 3.8 ms), our $d = 3$ masked + QRNG scheme incurs a 4× to 5.3× latency increase. This is acceptable for applications requiring <100 signatures per second.

Table 7: Performance comparison of the proposed masked SIS commitment vs. Kyber768 and Dilithium3 masked NTT multiplication on ARM Cortex-M4 at 7.3 MHz.

Operation	$d = 1$ (ms)	$d = 2$ (ms)	$d = 3$ (ms)
SIS commitment (ours, per-signature nonce)	0.8	3.2	7.2
Kyber768 basemul [3]	0.3	2.6	18.9
Dilithium3 basemul [3]	0.5	4.1	11.4

Table 8: Overhead breakdown for $d = 3$ signing (Cortex-M4, 7.3 MHz).

Operation	Time (ms)	Percentage of Total	Notes
QRNG seed acquisition	0.20–5.00	1%–33%	Device-dependent
CTR_DRBG mask generation	1.20	8%	$d \times 256$ AES calls
CTR_DRBG seed refresh	0.15	1%	AES-256 block encrypt
SIS commitment (masked)	7.20	48%	d^2 multiplications
W-OTS signing	4.50	30%	SHA3-256 + hash chains
Verkle path generation	2.10	14%	SIS proof generation
Total	15.15–20.15	100%	

7.3 Comparison with NIST FIPS 204 (ML-DSA)

The finalized NIST FIPS 204 standard (ML-DSA) represents the primary lattice-based post-quantum signature benchmark. Table 9 compares our Verkle-SIS scheme with ML-DSA-44 (Level 2, 128-bit security) across key metrics.

Table 9: Comparison of proposed Verkle-SIS scheme with NIST FIPS 204 ML-DSA-44.

Metric	ML-DSA-44 (FIPS 204)	Proposed Verkle-SIS (This Work)
Core operation	NTT polynomial multiplication (n = 256)	SIS vector commitment (d = 256)
Public key size	1312 bytes	$n \cdot \log_2 q = 256 \cdot 23 = 736$ bytes
Signature size	2420 bytes	≈8–12 KB (stateful, depends on h, k)
Signing time (Cortex-M4)	≈0.5 ms (unmasked)	7.2 ms (masked d = 3)
Verification time	≈0.3 ms	≈1.5 ms
Energy per signature (est.)	≈15 µJ at 3.3 V	≈200 µJ (dominated by masking)
Side-channel resistance (3rd-order CPA)	≈700 traces (from [3])	≥23,100 traces (this work)
Stateful	No	Yes (W-OTS index)

Note: Energy estimates assume 7.3 MHz operation at 3.3 V with typical Cortex-M4 current draw (≈15 mA active). Signature size for Verkle-SIS is approximate; exact value depends on branching factor $k = 16$, height $h = 8$, and SIS proof size ≈1.5 KB per path element.

While our scheme incurs higher signature size and signing latency due to masking and stateful W-OTS keys, it provides substantially stronger resistance against physical side-channel attacks—a critical advantage for deployment in adversarial physical environments (e.g., payment terminals, secure elements).

As a stateful scheme, our Verkle-SIS signature requires maintaining a 288-byte state (256-bit current seed + 4-byte leaf index). The public key is 736 bytes; a typical signature is 9408 bytes (see Table 9 for full

breakdown). This compares with ML-DSA-44's 1312-byte public key and 2420-byte signature, albeit with significantly higher side-channel resistance.

7.4 Practical Deployment Considerations

The IDQ Quantis PCIe is well-suited for server-class deployments (e.g., certificate authorities, blockchain validators) where PCI-Express slots are available. The IDQ Quantum Appliance provides a network interface for cloud or data-center deployments without client-side hardware requirements. The CryptoLabs USB QRNG is the most portable option, suitable for IoT gateways and edge devices running the Verkle signature scheme with a USB host port. All three have been successfully integrated with our CTR_DRBG implementation on the target platform via a simple seed-injection API.

System-Level QRNG Integration. For server-class deployments, the ID Quantique Quantis PCIe card is installed as a standard PCIe device; the Linux kernel driver provides a /dev/quantis character device from which 256-bit seeds are read via read() syscalls. For the ID Quantique Quantum Appliance, seeds are obtained via a REST API over TLS 1.3 (e.g., GET /api/v1/random/256). The CryptoLabs USB QRNG appears as a CDC-ACM serial device; seeds are read as raw binary data at 10 Mbit/s. In all cases, our CTR_DRBG implementation calls a hardware abstraction layer function qrng_get_seed(uint8_t *buf, size_t len) which blocks until the requested entropy is obtained. The measured overhead of this function call averages T_{PCIe} = 0.2 ms for the PCIe device, T_{USB} = 1.5 ms for the USB device, and T_{QA} = 5 ms (network latency) for the Quantum Appliance.

To prevent software-side snooping of the QRNG seed, we implement explicit memory sanitization immediately after seed injection. The function qrng_get_seed() reads the seed into a 32-byte buffer, passes it to aes256_ctr_drbg_init(), which copies the seed into the internal AES key schedule, and then explicitly overwrites the buffer with zeros using memset_s(buf, len, 0) before returning. This prevents the seed from remaining in stack or heap memory after initialization. Additionally, the CTR_DRBG internal state (256-bit key and 128-bit counter) is stored in a protected memory region that is zeroized during system reset or on detection of tampering. The AES key schedule is stored in registers where possible, and the stack is cleared after each signing operation using explicit_bzero() on all temporary variables. These practices follow the guidance of NIST SP 800-57 Part 1 regarding key zeroization and secure memory management.

However, it should be noted that while the QRNG provides *information-theoretic* security for the seed generation step, the subsequent CTR_DRBG key evolution relies on the computational hardness of AES. Therefore, the overall forward secrecy guarantee of the scheme is computational, not information-theoretic, against a computationally unbounded adversary.

Health Testing and Entropy Monitoring. Each QRNG device implements continuous health tests compliant with AIS-31 or NIST SP 800-90B. The IDQ Quantis PCIe performs startup tests (repetition count test, adaptive proportion test) and continuous online tests. We additionally implement vendor-independent health monitoring in our firmware: before each CTR_DRBG seed update, we invoke a non-parametric entropy estimation using the most recent 1024 output bits (the chi-square goodness-of-fit test at $\alpha = 0.01$). If the test fails (i.e., $p < 0.01$), the system falls back to a CSPRNG seeded from a hardware TRNG (STM32 RNG) and raises an alert. This fallback ensures availability even if the QRNG hardware degrades.

Practical Considerations for Embedded Use. The CryptoLabs USB QRNG requires a USB host port, making it suitable for IoT gateways and developer boards (e.g., Raspberry Pi, BeagleBone). For deeply embedded microcontrollers without USB, we recommend the STM32 hardware RNG (which provides $H_{\min} \geq 128$ bits as per [ST Application Note 4230]) as a lower-cost alternative; the security parameter $\lambda = 128$ is then used accordingly. However, it should be noted that switching from the QRNG to the STM32 hardware

RNG introduces a theoretical security downgrade: the loss of provable quantum non-determinism eliminates the strict forward-secrecy guarantee against computationally unbounded adversaries. While the STM32 RNG provides sufficient entropy for classical security and is NIST-compliant, it is a classical chaotic entropy source that could, in principle, be predicted given complete knowledge of the physical state. The QRNG, by contrast, provides information-theoretic security for the seed generation step, ensuring that even an unbounded adversary cannot predict the seed. For applications requiring the highest level of forward secrecy, the QRNG should be retained; for standard commercial deployments, the hardware TRNG is acceptable. The additional power consumption of the USB QRNG is approximately $P = V \times I = 5 \times 0.01 = 0.05$ W = 50 mW, which is acceptable for line-powered gateways but may be prohibitive for battery-powered sensors.

For fully embedded deployments (microcontrollers without USB or PCIe), a ring-oscillator TRNG or an on-chip entropy source (e.g., STM32 hardware RNG) can substitute as the seed source at reduced min-entropy. In this case, the security parameter λ must be adjusted to reflect $H_{\min}$ of the on-chip source, typically $\gamma \geq 128$ bits for NIST-compliant on-chip TRNGs.

For applications requiring more than 2^{32} signatures, the tree height h can be increased (e.g., $h = 10$ yields $16^{10} = 2^{40}$ leaves). This increases authentication path length (h proofs) and verification time proportionally. Signing time remains unchanged (only leaf index selection). Storage for the tree can be optimized by generating leaves on the fly using the CTR_DRBG's forward-secure property, trading computation for memory.

The masked SIS commitment consists of $d \times d$ modular multiplications. Each multiplication requires $O(\log q)$ bit operations. On an FPGA, these can be pipelined; we estimate a $d = 3$ implementation would require approximately 12,000 LUTs (Virtex-7) and 64 DSP slices for 256-bit Montgomery multipliers. This is comparable to a masked NTT unit.

7.5 Full-System Performance Metrics

Table 10 reports the complete performance profile of our Verkle-SIS signature scheme on the STM32F303 target (Cortex-M4, 7.3 MHz) with $d = 3$ masking and $k = 16$, $h = 8$ (2^{32} one-time keys).

Table 10: Full-system performance breakdown for proposed Verkle-SIS scheme ($d = 3$).

Component	Time (ms)	Memory (Bytes)	Notes
Key Generation (one-time)			
QRNG seed acquisition	0.2–5.0	256	Depends on device (PCIe: 0.2, USB: 1.5, Net: 5)
CTR_DRBG initialization	0.15	64	AES-256 key schedule
Root tree construction	418.0	8192	Precomputation of all k^h leaves (offline)
Signing (per operation)			
CTR_DRBG seed refresh	0.15	32	
Mask share generation ($256 \times d$ values)	1.20	2048	Fresh randomness per signature
SIS commitment (masked)	7.20	4096	Main leakage target (Table 6)
W-OTS signing (SHA3-256)	4.50	1024	$t = 67$, $n = 256$

(Continued)

Table 10 (continued)

Component	Time (ms)	Memory (Bytes)	Notes
Verkle path generation (h = 8 proofs)	2.10	3072	Each SIS proof ≈ 384 bytes
Total signing	**15.15**	**10,560**	
Verification (per signature)			
W-OTS verification	3.80	512	
SIS proof verification (8×)	8.40	3072	
Root comparison	0.05	32	
Total verification	**12.25**	**3616**	
Storage			
Public key (c_{root})	—	736	$n \cdot \log_2 q = 256 \cdot 23$ bits
Signature (typical)	—	9408	s (4 B) + sign (6 KB) + Y_s (1 KB) + 8 proofs (3 KB)
State (current seed + index)	—	288	256 B seed + 4 B counter

Note: Memory values represent peak stack + heap usage during each operation. Tree construction (418 ms) is a one-time cost amortized over all signatures. For applications requiring frequent reboots, the root tree can be stored in flash (≈8 KB per 2^{16} subtree).

8 Conclusion

This paper has presented a side-channel-resistant post-quantum digital signature scheme based on Verkle trees and SIS-based vector commitments, with formal EUF-CMA security in the quantum random oracle model. The scheme's structural avoidance of NTT-based polynomial multiplication eliminates the primary attack surface exploited by recent higher-order CPA and single-trace belief propagation attacks against ML-KEM and ML-DSA. Experimental measurements on ChipWhisperer Nano and Husky platforms confirm that third-order masked SIS commitments require at least 23,100 power traces to reach 50% key recovery success rate, compared with 700–2200 traces for equivalently masked Kyber and Dilithium implementations.

The integration of ID Quantique Quantis PCIe, ID Quantique Quantum Appliance, and CryptoLabs USB QRNG hardware as entropy sources was characterized using NIST SP 800-90B, confirming $H_{\min} \geq 0.998$ bits/bit across all devices. The formal proof incorporates this measurement as a concrete bound, tightening the security claim beyond what is achievable with classical PRNG seeding. An experiment with a low-entropy LCG seed confirmed that seed quality is a directly measurable contributor to physical SCA resistance, not merely a theoretical concern.

Future work will explore hardware acceleration of the masked SIS commitment on FPGA, extending the scheme to larger tree heights ($h > 10$) with parallelized batch verification, and formally analyzing the scheme's resistance to fault injection attacks on the CTR_DRBG seed management.

Acknowledgement: The authors express their gratitude to the staff of the Department of Computer Science, Caucasus University, and the Information Security Laboratory of the Institute of Information and Computational Technologies for their assistance.

Funding Statement: The research work was funded by the Ministry of Science and Higher Education of Kazakhstan and carried out within the framework of the project AP23488112 "Development and Study of a Quantum-Resistant Digital Signature Scheme Based on a Verkle Tree" at the Institute of Information and Computational Technologies.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization: Maksim Iavich and Nursulu Kapalova; methodology: Maksim Iavich; software: Kunbolat Algazy; validation: Nursulu Kapalova; formal analysis: Kunbolat Algazy; investigation: Maksim Iavich and Nursulu Kapalova; resources: Nursulu Kapalova; data curation: Maksim Iavich and Kunbolat Algazy; writing—original draft preparation: Maksim Iavich; writing—review and editing: Nursulu Kapalova; visualization: Maksim Iavich; supervision: Kunbolat Algazy; project administration: Kunbolat Algazy; funding acquisition: Nursulu Kapalova. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The source code for the masked SIS commitment implementation on the ARM Cortex-M4 platform, along with the CPA/HOCPA analysis scripts, is available from the corresponding author upon reasonable request. Raw ChipWhisperer power traces—captured across all masking orders (d = 1 to d = 4), with a maximum budget of 80,000 traces per experiment (used for the unsuccessful d = 4 attack; the successful d = 3 threshold was 23,100 traces)—and QRNG entropy samples (10^8 bits per device) used for the NIST SP 800-90B characterization are also available upon reasonable request, subject to storage capacity limitations.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. FIPS 204. Module-lattice-based digital signature standard. Gaithersburg, MD, USA: Federal Information Processing Standards Publication; 2024.
2. Kocher P, Jaffe J, Jun B. Differential power analysis. In: Wiener M, editor. Proceedings of the Advances in Cryptology—CRYPTO'99; 1999 Aug 15–19; Santa Barbara, CA, USA. Berlin/Heidelberg, Germany: Springer; 1999. p. 388–97, doi:10.1007/3-540-48405-1_25.
3. Tosun T, Oswald E, Savaş E. Non-profiled higher-order side-channel attacks against lattice-based post-quantum cryptography. 2025 [cited 2026 Apr 30]. Available from: https://eprint.iacr.org/2025/1257.
4. Primas R, Pessl P, Mangard S. Single-trace side-channel attacks on masked lattice-based encryption. In: Proceedings of the International Conference on Cryptographic Hardware and Embedded Systems; 2017 Sep 25–28; Taipei, Taiwan. Cham, Switzerland: Springer International Publishing. 2017. p. 513–33. doi:10.1007/978-3-319-66787-4_25.
5. Iavich M, Kuchukhidze T. Investigating CRYSTALS-Kyber vulnerabilities: attack analysis and mitigation. Cryptography. 2024;8(2):15. doi:10.3390/cryptography8020015.
6. Iavich M, Kapalova N. Optimizing post-quantum digital signatures with Verkle trees and quantum seed-based pseudo-random generators. Computers. 2025;14(3):103. doi:10.3390/computers14030103.
7. Iavich M, Kuchukhidze T, Bocu R. A post-quantum digital signature using Verkle trees and lattices. Symmetry. 2023;15(12):2165. doi:10.3390/sym15122165.
8. Kuszmaul J. MIT PRIMES research report. Cambridge, MA, USA: Massachusetts Institute of Technology. 2019 [cited 2026 Mar 5]. Available from: https://math.mit.edu/research/highschool/primes/materials/2018/Kuszmaul.pdf.
9. Bernstein DJ. Introduction to post-quantum cryptography. In: Bernstein DJ, Buchmann J, Dahmen E, editors. Post-quantum cryptography. Berlin/Heidelberg, Germany: Springer; 2009. p. 1–14. doi:10.1007/978-3-540-88702-7_1.
10. Libert B, Ling S, Nguyen K, Wang H. Zero-knowledge arguments for lattice-based accumulators: logarithmic-size ring signatures and group signatures without trapdoors. J Cryptol. 2023;36(3):23. doi:10.1007/s00145-023-09470-6.
11. de Castro L, Peikert C. Functional commitments for all functions, with transparent setup and from SIS. In: Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques; 2023 Apr 23–27; Lyon, France. Cham, Switzerland: Springer Nature; 2023. p. 287–320. doi:10.1007/978-3-031-30620-4_10.

12. NIST SP 800-90A Rev. 1. Recommendation for random number generation using deterministic random bit generators. Gaithersburg, MD, USA: National Institute of Standards and Technology; 2015.
13. Shang T, Chen R, Lei Q. Quantum random oracle model for quantum public-key encryption. IEEE Access. 2019;7:130024–31. doi:10.1109/ACCESS.2019.2940406.
14. Fei Y, Ding AA, Lao J, Zhang L. A statistics-based fundamental model for side-channel attack analysis. 2014 [cited 2026 Jan 1]. Available from: https://eprint.iacr.org/2014/152.
15. Prouff E, Rivain M, Bevan R. Statistical analysis of second order differential power analysis. IEEE Trans Comput. 2009;58(6):799–811. doi:10.1109/TC.2009.15.
16. Dubrova E, Ngo K, Gärtner J, Wang R. Breaking a fifth-order masked implementation of CRYSTALS-Kyber by copy-paste. In: Proceedings of the 10th ACM Asia Public-Key Cryptography Workshop; 2023 Jul 10–14; Melbourne, VIC, Australia. New York, NY, USA: ACM; 2023. p. 10–20. doi:10.1145/3591866.3593072.
17. Reparaz O, Roy SS, de Clercq R, Vercauteren F, Verbauwhede I. Masking ring-LWE. J Cryptogr Eng. 2016;6(2):139–53. doi:10.1007/s13389-016-0126-5.
18. Whitnall C, Oswald E. A fair evaluation framework for comparing side-channel distinguishers. J Cryptogr Eng. 2011;1(2):145–60. doi:10.1007/s13389-011-0011-1.
19. Yan Y, Oswald E, Roy A. Not optimal but efficient: a distinguisher based on the Kruskal-Wallis test. In: Proceedings of the 26th International Conference on Information Security and Cryptology; 2023 Nov 29–Dec 1; Seoul, Republic of Korea. Cham, Switzerland: Springer; 2024. p. 240–58. doi:10.1007/978-981-97-1235-9_13.