# Post-Quantum Online/Offline Signatures

Martin R. Albrecht, Nicolas Gama, James Howe, and Anand Kumar Narayanan

SandboxAQ, Palo Alto, CA, USA,
{firstname.secondname}@sandboxaq.com

**Abstract.** Post-quantum signatures have high costs compared to RSA and ECDSA, in particular for smart cards. A line of work originating from Even, Goldreich, and Micali (CRYPTO'89) aimed to reduce digital signature latency by splitting up signing into an *online* and *offline* phase. The online/offline paradigm combines an ordinary long-term signature scheme with a fast, generally one-time, signature scheme. We reconsider this paradigm in the context of lattice-based post-quantum signatures in the GPV framework, with an example instantiation based on Falcon.

## 1 Introduction

In July 2022, NIST [AAC+22] announced its first set of post-quantum cryptography schemes intended for standardisation: the key encapsulation mechanism (KEM) Kyber (aka ML-KEM in FIPS 203 [ML-23b]), and three digital signature schemes, Dilithium (aka ML-DSA in FIPS 204 [ML-23a]), SPHINCS$^+$ (aka SLH-DSA in FIPS 205 [SLH23]), and lastly Falcon (aka FN-DSA) [PFH+22]. While Kyber has *computational* costs similar to RSA and elliptic curves, all three signature standards/standardisation candidates compare less favourably.

Online/offline signatures were formally introduced by Even, Goldreich and Micali [EGM90,EGM96] as a way to speed-up the signing process in applications with constrained computing resources or where signature latency is critical, such as in smart cards. The construction relies on (fast) one-time digital signature algorithms as well as standard (slower) digital signature algorithms.

The basic idea is to break-up signature generation into two parts, one of which is a pre-computation (*offline*), independent of any message $m$ to be signed. This pre-computation can be executed during idle time and consists of the generation of a one-time public-key and secret-key pair, where the one-time public-key is signed using the long-term secret-key. This signing operation will be more computationally expensive than the one-time signature scheme. The *online* phase then depends on and signs $m$, using the one-time signature scheme with the one-time keys as input.

Among the signature schemes selected, Falcon was selected by NIST for its more compact signatures and fast verification: "due to its low bandwidth and fast verification, Falcon may be a superior choice in some constrained protocol scenarios" [AAC+22]. However, for general-purpose constrained applications, Falcon does not seem the appropriate choice, as the same report notes. This is in

addition to its reliance on double-precision floating-point arithmetic for fast implementation, which can potentially cause issues with signature correctness and which adds an additional attack vector [KA21,GMRR22,ZLYW23,HW23,LTYZ24].

Falcon's genealogy dates back to the earliest proposals for lattice-based signatures, including GGH [GGH97] and NTRUSign [HHP+03]. The general idea of these signatures is that the verification key consists of a description of a lattice and the signing key consists of a trapdoor, allowing to sample short elements in the lattice. This then allows to find vectors in the span of the lattice that are close to some target vector produced by calling $H(m)$, where $H(\cdot)$ is some hash function, modelled as a Random Oracle. As noted in [NR06], care must be taken that these close vectors do not leak information about the trapdoor. Indeed, in [NR06] devastating attacks were presented against both GGH and NTRUSign, recovering the secret trapdoor using statistical methods on the signatures.

With the GPV framework [GPV08], a method was proposed to produce signatures that are statistically close to being independent of the trapdoor, leading to a signature scheme framework with provable guarantees. Later, NTRUSign was improved [SS11] and also used to build an identity-based encryption scheme [DLP14]. With further improvements [DP16], the groundwork was laid for a practical and secure hash-and-sign lattice-based signature scheme, producing Falcon. Proposals to improve upon Falcon were subsequently proposed [EFG+22,DPPvW22].

Carefully sampling from a distribution essentially independent of the trapdoor as in [GPV08] requires noticeable computational resources and typically floating-point arithmetic, leading to the "downsides" noted above.

Given that the attacks from [NR06] mentioned above are statistical, it is natural to ask how many signatures are required to mount them. Put differently, in an online/offline paradigm where we instantiate the one-time signature scheme with (essentially) NTRUSign, how many signatures can be produced before security no longer holds?

To tackle this question, we rely on statistical tools introduced to the area of lattice-based cryptography in [BLR+18]: the Rényi divergence.

### 1.1 Contributions

We revisit the idea of online/offline signatures for the post-quantum era and propose a simple and efficient signature scheme, along the lines of GPV, which is partially interoperable with Falcon.

We provide a security proof for the proposed signature scheme, quantifying the security as a function of the number of online signatures furnished. This means that our scheme guarantees security only if a tiny number of online signatures allowed before refreshing in the offline phase. In particular, this implies that statistical/geometric attacks such as [NR06,DN12] that broke NTRUSign will not infer anything significant about the ephemeral secret key from seeing these few online signatures. A key ingredient in our security proof is Rényi divergence analysis as in [BLR+18]. En route, we present new bounds for the Rényi

divergence of a discrete spherical Gaussian and a discrete spherical Gaussian convolved (of possibly different mean and covariance) with a compact distribution. This is implicit in the proof of Lemma 1 and may be of use in other cryptographic contexts. We derive these new bounds from first principles since the standard Rényi divergence bounds from [BLR+18] and its extensions do not quite apply.

We adapt our scheme to the Falcon setting and show that the scheme remains secure, by quantifying the security in terms of the number of online signatures. Design choices such as lattice dimension, modulus, etc. conform to Falcon-512, enabling easy adoption. Further, key generation and signature verification remain unchanged from Falcon, except for an increase in the allowed verification threshold norm of the signatures. This allows the long-term and few-time signature schemes to share modules, reducing footprint and benefiting implementations on constrained or embedded devices. The signature scheme can be used in different environments, ranging from one-time use up to eight uses. We provide performance expectations based on the security needs of the application within these ranges. As expected, our implementation provides good performance in the online phase when considering a floating point implementation. We consider this permissible despite potential side-channel attack concerns because of the low number of signing operations performed per key.

### 1.2 Organisation

In Section 2, we build up the required mathematical notation and briefly describe the cryptographic definitions and primitives used. In Section 3, we present the online/offline signature scheme based on GPV signatures and prove its security. Parts of the proof are deferred to Section 4.1. In Section 4, we tailor our online/offline signature scheme to Falcon and derive concrete security guarantees in terms of the number of online signatures. In Section 4.1, we detail the implementation of our Falcon-based online/offline scheme and present benchmarks to highlight its practicality.

## 2 Preliminaries

**Lattices.** We write vectors as columns vectors and use boldface latin alphabets, such as $\mathbf{u}$ or $\mathbf{v}$. The $i$-th coordinate of a vector $\mathbf{u}$ will be denote as $u_i$. Matrices will be written with boldface capital latin alphabets such as $\mathbf{A}, \mathbf{B}$. We write $\mathbb{I}_n$ for the identity matrix in dimension $n$. In our context, a lattice is a discrete free $\mathbb{Z}$-submodule of a finite dimensional real vector space, endowed with the Euclidean metric. The dimension of a lattice is its rank as a $\mathbb{Z}$-module. Lattices we encounter will be explicitly presented in one of the following two ways. In ambient dimension $d$, an ordered basis $\mathbf{B} := (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n) \subset \mathbb{R}^d$ of $\mathbb{R}$-linearly independent vectors defines the rank $n$ lattice $\mathcal{L}(\mathbf{B}) := \{\sum_{i=1}^{n} z_i \cdot \mathbf{b}_i \mid \mathbf{z} \in \mathbb{Z}^n\} \subset \mathbb{R}^d$. The basis $\mathbf{B}$ will also be thought of as the matrix $\mathbf{B}$ consisting of $(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n)$ as its sequence of columns, and vice versa. Therefore, $\mathcal{L}(\mathbf{B})$ is the $\mathbb{Z}$-module generated by the columns of $\mathbf{B}$. The second way lattices are presented as sets. For example,

the dual of a lattice $\Lambda \subset \mathbb{R}^d$ is defined as $\Lambda^* \coloneqq \left\{\mathbf{u} \in \mathbb{R}^d \mid \langle \mathbf{u}, \mathbf{x} \rangle \in \mathbb{Z}, \forall \mathbf{x} \in \Lambda\right\} \subset \mathbb{R}^d$ where $\langle , \rangle$ is the inner product in $\mathbb{R}^d$. In either case, lattices are embedded in some real space $\mathbb{R}^d$, which endows them with the Euclidean metric.

For a positive odd integer $q$, let $\mathbb{Z}_q$ denote the ring $\mathbb{Z}/q\mathbb{Z}$ of residues modulo $q$ and we choose $\{-\lfloor \frac{q}{2} \rfloor, \ldots, 0, \ldots, \lfloor \frac{q}{2} \rfloor\} \subset \mathbb{Z}$ as a preferred set of representatives. For a vector $\mathbf{x} \in \mathbb{Z}^d$ with integer coordinates, let $\mathbf{x} \bmod q \in \mathbb{Z}_q^d$ denote the reduction of each coordinate modulo $q$.

**Babai's Rounding.** Babai's rounding algorithm [Bab86] is an algorithm to find a lattice point close to a given target point . This is efficient to compute given a "good" basis $\mathbf{G}$ for a lattice and a target vector $\mathbf{c}$, where $\mathbf{c}$ is in the column $\mathbb{R}$-span of $\mathbf{G}$. The algorithm proceeds by computing $\mathbf{G} \cdot \lfloor \mathbf{G}^{-1} \cdot \mathbf{c} \rceil$. In essence, writing the target $\mathbf{c} = \sum_{i=1}^{d} \alpha_i \cdot \mathbf{g}_i$ as a real linear combination of the columns of $\mathbf{G}$, the Babai rounding $\mathbf{G} \cdot \lfloor \mathbf{G}^{-1} \cdot \mathbf{c} \rceil = \sum_{i=1}^{d} \lfloor \alpha_i \rceil \cdot \mathbf{g}_i$ is the lattice vector obtained by rounding the real coordinates to the nearest integers $\lfloor \alpha_i \rceil$.

**Babai's Nearest Plane.** As before, consider a good basis $\mathbf{G}$ and a target vector $\mathbf{c} \in \mathbb{R}^d$ in the column $\mathbb{R}$-span of $\mathbf{G}$. Babai's nearest plane algorithm [Bab86] starts at the $d$-th dimension and asks for a lattice vector $\mathbf{v}_d$, which by addition shifts the real hyperplane $H_{\mathbf{G}}^{d-1}$ generated by first $d-1$ columns of $\mathbf{G}$ closest to $\mathbf{c}$. To find $\mathbf{v}_d$, write the target $\mathbf{c} = \sum_{i=1}^{d} \alpha_i \cdot \mathbf{g}_i^*$ as a real linear combination of the Gram-Schmidt orthogonalisation vectors $\mathbf{g}_i^*$ of the columns of $\mathbf{G}$ and set $\mathbf{v}_d \leftarrow \lfloor \alpha_d \rceil \cdot \mathbf{g}_d$, where $\lfloor \alpha_d \rceil$ is the nearest integer to $\alpha_d$. The solution for the nearest plane algorithm is $\sum_{i=1}^{d} \mathbf{v}_i$, where one solves for $\mathbf{v}_{d-1}$ recursively, as follows. Forget the last column of $\mathbf{G}$ and consider the same problem, now in one fewer dimension with the new target $\sum_{i=1}^{d-1} \alpha_i \cdot \mathbf{g}_i^* + \lfloor \alpha_d \rceil \cdot \mathbf{g}_d^* - \lfloor \alpha_d \rceil \cdot \mathbf{g}_d$. Here, $\sum_{i=1}^{d-1} \alpha_i \cdot \mathbf{g}_i^* + \lfloor \alpha_d \rceil \cdot \mathbf{g}_d^*$ is the orthogonal projection of $\mathbf{c}$ onto $\lfloor \alpha_d \rceil \cdot \mathbf{g}_d + H_{\mathbf{G}}^{d-1}$. The closeness guarantee of the output to the target is better for the nearest planes compared to rounding. There are also randomised versions of the nearest plane algorithm, including fast versions applicable to special lattices [DP16]. But rounding is much faster, in comparison with both the plain and the specialised randomised versions.

**NTRU Lattices.** Let $n = 2^a$ denote a positive power of 2. The NTRU lattices underlying Falcon are certain rank 2 modules over the ring of integers $\mathcal{O}_n \cong \mathbb{Z}(X)/\langle X^n + 1 \rangle$ of the 2n-th cyclotomic field $\mathbb{Q}(X)/\langle X^n + 1 \rangle$, embedded in an inner product space over $\mathbb{R}(X)/\langle X^n + 1 \rangle$, generated as follows. Let $q$ be a prime number that splits in the extension $\mathcal{O}_n/\mathbb{Z}$. That is, $q = 1 \bmod \varphi(n)$, where $\varphi(n) = 2^a - 2^{a-1}$ is the Euler's totient. Start with a secret key consisting of $f, g, F, G \in \mathcal{O}_N$ such that $f \cdot G - g \cdot F = q$, $f$ modulo $q\mathcal{O}_n$ is a unit and integer vectors corresponding to the representations of $f, g, F, G$ are short. Short enough means that the matrix $\mathbf{B} \coloneqq \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}^t$ has Gram-Schmidt norm at most a constant times $\sqrt{q}$, and is the secret trapdoor basis. When instantiating using

Falcon parameters, this constant will be 1.17. The NTRU lattice associated with the secret key $f, g, F, G$ is the rank 2 $\mathcal{O}_n$-module generated by $\mathbf{B}$. The public key is then set as the unique $h \in \mathcal{O}_n$ satisfying $h = g \cdot f^{-1} \mod q\mathcal{O}_n$. Then the same NTRU lattice is also generated by $\begin{bmatrix} 1 & h \\ 0 & q \end{bmatrix}^t$, except, this public basis of the same lattice is not short by design. The public and private bases can both be represented succinctly as elements in the ring $\mathcal{O}_n$, which can be thought of as polynomials. In fact, Falcon represents the trapdoor in the Fourier basis as detailed Section 2.5, facilitating fast signing times. We will build our signature scheme in the language of lattices over $\mathbb{Z}$, and then specialize to NTRU lattices. Therefore, it is conceptually helpful to also think of the NTRU lattices, which are $\mathcal{O}_n$-modules as lattices over $\mathbb{Z}$. To this end, associate with $-f$, the $n \times n$ integer matrix whose $i$-th row is the integer vector corresponding to $-f \cdot X^i \mod X^n + 1$. Likewise, for the other elements $g, -F, G$ and $h$. Under this correspondence, the NTRU lattice may also be viewed as a rank $2n$-lattice over $\mathbb{Z}$.

**Distributions.** For $\mathbf{c} \in \mathbb{R}^d$ and a positive $s \in \mathbb{R}$, let $\rho_{\mathbf{c},s} : \mathbb{R}^d \longrightarrow \mathbb{R}$ taking $\mathbf{x} \longmapsto \exp\left(-\pi\|\mathbf{x} - \mathbf{c}\|^2/s^2\right)$ denote the Gaussian function with parameter $s$ and centred at $\mathbf{c}$ . For a lattice $\Lambda \subset \mathbb{R}^d$, positive $s \in \mathbb{R}$ and vector $\mathbf{c} \in \mathbb{R}^d$, let $D_{\Lambda,s,\mathbf{c}}$ denote the spherical discrete Gaussian probability mass function with support $\Lambda$ and mass proportional to $D_{\Lambda,s,\mathbf{c}}(\mathbf{x}) \sim \rho(\mathbf{c}, s), \forall\, \mathbf{x} \in \Lambda$. That is,

$$D_{\Lambda,s,\mathbf{c}}(\mathbf{x}) = \frac{\rho_{\mathbf{c},s}(\mathbf{s})}{\sum_{\mathbf{x}\in\Lambda} \rho_{\mathbf{c},s}(\mathbf{x})}, \forall\, \mathbf{x} \in \Lambda.$$

When the discrete Gaussian is centered at the origin, we sometimes suppress the last subscript, that is, $\mathcal{D}_{\Lambda,s} := \mathcal{D}_{\Lambda,s,\mathbf{0}}$. For a lattice $\Lambda$ and a positive real $\epsilon$, the smoothing parameter $\eta_\epsilon(\Lambda)$ is the smallest positive real number $s$ such that $\sum_{\mathbf{s}\in\Lambda\setminus\mathbf{0}} \rho(\mathbf{s}, 1/s) \geq \epsilon$ [MR04].

*Rényi Divergence.* Following [BLR+18], the order $a \in (1, \infty]$ Rényi divergence of a pair $(\mathcal{P}, \mathcal{Q})$ of discrete distributions with $\mathrm{Supp}(\mathcal{P}) \subseteq \mathrm{Supp}(\mathcal{Q})$ is defined as

$$R_a(\mathcal{P}||\mathcal{Q}) := \left( \sum_{x\in\mathrm{Supp}(\mathcal{P})} \frac{\mathcal{P}(x)^a}{\mathcal{Q}(x)^{a-1}} \right)^{\frac{1}{a-1}}. \tag{1}$$

We use the following two properties of Rényi divergence in our security proofs.

- Probability preservation: For every event $E \subseteq \mathrm{Supp}(\mathcal{P})$ and $a \in (1, \infty]$

$$\mathcal{Q}(E) \geq \frac{\mathcal{P}(E)^{\frac{a}{a-1}}}{R_a\left(\mathcal{P}||\mathcal{Q}\right)}. \tag{2}$$

- Multiplicativity: The Rényi divergence of a pair of product distributions is the product of the Rényi divergences, that is,

$$R_a\left(\prod_i \mathcal{P}_i || \prod_i \mathcal{Q}_i\right) = \prod_i R_a(\mathcal{P}_i||\mathcal{Q}_i). \tag{3}$$

### 2.1 Digital Signatures

**Definition 1 (Signature Scheme).** *A signature scheme $\Sigma$ consists of three PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *such that:*

$\mathsf{KeyGen}$ *The key generation algorithm is a randomised algorithm that takes as input a security parameter $1^\lambda$ and outputs a pair $(\mathsf{vk}, \mathsf{sk})$, the* verification key *and* signing key*, respectively. We write $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.*

$\mathsf{Sign}$ *The signing algorithm takes as input a signing key $\mathsf{sk}$ and a message $\mu$ and outputs a signature $\sigma$. We write this as $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mu)$. The signing algorithm may be randomised or deterministic. We may write $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mu; r)$ to unearth the used randomness explicitly.*

$\mathsf{Verify}$ *The verification algorithm takes as input a verification key $\mathsf{vk}$, a signature $\sigma$ and a message $\mu$ and outputs a bit $b$, with $b = 1$ meaning the signature is valid and $b = 0$ meaning the signature is invalid.* $\mathsf{Verify}$ *is a deterministic algorithm. We write $b \leftarrow \mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$.*

*We require that except with negligible probability over $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, it holds that $\mathsf{Verify}(\mathsf{vk}, \mathsf{Sign}(\mathsf{sk}, \mu), \mu) = 1$ for all $\mu$.*

We rely on the standard notion of existential unforgeability under chosen message attacks:

**Definition 2 (EUF-CMA).** *We define*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma}(\lambda) \coloneqq \Pr[\text{EUF-CMA}^{\mathcal{A}}_{\Sigma}(\lambda) \Rightarrow 1]$$

*for* $\text{EUF-CMA}^{\mathcal{A}}_{\Sigma}(\lambda)$ *as in Figure 1 (with $Q > \mathsf{poly}(\lambda)$) and say a signature scheme $\Sigma$ is EUF-CMA secure if no PPT/BQP adversary $\mathcal{A}$ has non-negligible advantage* $\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma}(\lambda)$.

### 2.2 Few-Times Signatures

We will rely on "few-times signature" schemes, which are signature schemes that promise unforgeability in the presence of a signing oracle supporting up to $Q$ signatures.

**Definition 3 (EUF-CMA$_Q$).** *We define*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma,Q}(\lambda) \coloneqq \Pr[\text{EUF-CMA}^{\mathcal{A}}_{\Sigma,Q}(\lambda) \Rightarrow 1]$$

*for* $\text{EUF-CMA}^{\mathcal{A}}_{\Sigma,Q}(\lambda)$ *as in Figure 1 and say a signature scheme $\Sigma$ is EUF-CMA secure for $Q$ queries if no PPT/BQP adversary $\mathcal{A}$ has non-negligible advantage* $\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma,Q}(\lambda)$.

| $\text{EUF-CMA}^{\mathcal{A}}_{\Sigma,Q}(\lambda)$ | $\mathsf{SIGN}(\mu)$ |
|---|---|
| $\mathcal{Q}, c \leftarrow \emptyset, 0$ | **if** $c \geq Q$ **then abort** |
| $\mathsf{vk}, \mathsf{sk} \leftarrow \Sigma.\mathsf{KeyGen}(1^\lambda)$ | $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mu)$ |
| $(\mu^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathsf{SIGN}}(\mathsf{vk})$ | $c \leftarrow c + 1$ |
| **return** $(\mu^\star, \cdot) \notin \mathcal{Q} \wedge \mathsf{Verify}(\mathsf{vk}, \sigma^\star, \mu^\star)$ | $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$ |
| | **return** $\sigma$ |

**Fig. 1.** Existential Unforgeability under Chosen Message Attacks Few Times Signature Schemes (EUF-CMA$_Q$). EUF-CMA is recovered by setting $Q > \mathsf{poly}(\lambda)$.

### 2.3 Online/Offline Signatures

The notion of online/offline signatures was first proposed by Even, Goldreich, and Micali in [EGM90,MGE91], and improved upon in later works [EGM96,ST01]. The online/offline paradigm aims to offload significant amounts of computation to a pre-computation (or offline) phase that does not depend on the message and can thus be precomputed.

**Definition 4 (Online/Offline Signature Scheme).** *An online-offline signature scheme* $\Sigma$ *consists of four PPT algorithms* $(\mathsf{KeyGen}, \mathsf{PreSign}, \mathsf{Sign}, \mathsf{Verify})$ *such that:*

KeyGen *The key generation algorithm is a randomised algorithm that takes as input a security parameter* $1^\lambda$ *and outputs a pair* $(\mathsf{vk}, \mathsf{sk})$*, the* verification key *and* signing key*, respectively. We write* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

PreSign *The pre-signing algorithm takes as input the signing key* sk *and outputs a pre-signing state* $\rho$*. The presigning algorithm is randomised. We write this as* $\rho \leftarrow \mathsf{PreSign}(\mathsf{sk})$*. The signing algorithm may be randomised or deterministic. We may write* $\rho \leftarrow \mathsf{PreSign}(\mathsf{sk}; r)$ *to unearth the used randomness explicitly.*

Sign *The signing algorithm takes as input a signing key* sk*, a pre–signing stage* $\rho$ *and a message* $\mu$ *and outputs a signature* $\sigma$*. We write this as* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \rho, \mu)$*. The signing algorithm may be randomised or deterministic. We may write* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \rho, \mu; r)$ *to unearth the used randomness explicitly.*

Verify *The verification algorithm takes as input a verification key* vk*, a signature* $\sigma$ *and a message* $\mu$ *and outputs a bit* $b$*, with* $b = 1$ *meaning the signature is valid and* $b = 0$ *meaning the signature is invalid.* Verify *is a deterministic algorithm. We write* $b \leftarrow \mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$.

*We require that except with negligible probability over* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and over* $\rho \leftarrow \mathsf{PreSign}(\mathsf{sk})$*, it holds that* $\mathsf{Verify}(\mathsf{vk}, \mathsf{Sign}(\mathsf{sk}, \rho, \mu), \mu) = 1$ *for all* $\mu$*.*

Online-offline signature schemes can be constructed generically by composing two signature schemes. A regular long-term signature scheme, for the offline phase, and a one-time or few-time signature scheme, for both online and offline

| $\mathsf{KeyGen}(1^\lambda)$ | $\mathsf{PreSign}(\mathsf{sk})$ | $\mathsf{Sign}(\mathsf{sk}, \rho, \mu)$ |
|---|---|---|
| $\mathsf{vk}, \mathsf{sk} \leftarrow \Sigma.\mathsf{KeyGen}(1^\lambda)$ | $\mathsf{vk}_Q, \mathsf{sk}_Q \leftarrow \Sigma_Q.\mathsf{KeyGen}(1^\lambda)$ | **assert** $\rho.c < Q$ |
| **return** $\mathsf{vk}, \mathsf{sk}$ | $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{vk}_Q)$ | $\sigma_Q \leftarrow \Sigma_Q.\mathsf{Sign}(\rho.\mathsf{sk}_Q, \mu)$ |
| | $c \leftarrow 0$ | $\rho.c \leftarrow \rho.c + 1$ |
| | **return** $\rho \coloneqq (\mathsf{vk}_Q, \mathsf{sk}_Q, \sigma, c)$ | **return** $\sigma \coloneqq (\rho.\mathsf{vk}_Q, \rho.\sigma, \sigma_Q)$ |

$\mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$

**return** $\Sigma.\mathsf{Verify}(\mathsf{vk}, \sigma, \mathsf{vk}_Q) \wedge \Sigma_Q.\mathsf{Verify}(\mathsf{vk}_Q, \sigma_Q, \mu)$

**Fig. 2.** A generic online/offline signature scheme construction.

phases. We will denote the former by $\Sigma$ and the latter by $\Sigma_Q$. We give the generic construction in Figure 2.

The security notion for online-offline signature schemes is identical to the EUF-CMA notion for general signature schemes except for semantic changes. In particular, the adversary has oracle access to both $\mathsf{Sign}$ and $\mathsf{PreSign}$ and its advantage is bounded as follows.

**Proposition 1.** *Let $\Sigma'$ be the online-offline construction in Figure 2. Then for any PPT $\mathcal{A}$, we have $2 \cdot \mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma'}(\lambda) \leq \mathsf{Adv}^{\text{euf-cma}}_{\mathcal{B},\Sigma}(\lambda) + Q' \cdot \mathsf{Adv}^{\text{euf-cma}}_{\mathcal{C},\Sigma_Q,Q}(\lambda)$, where $\mathcal{B}$ and $\mathcal{C}$ are PPT adversaries against $\Sigma$ and $\Sigma_Q$ respectively and where $Q'$ is the total number of queries made by $\mathcal{A}$.*

*Proof (Sketch).* The verification algorithm checks $\Sigma.\mathsf{Verify}(\mathsf{vk}, \sigma, \mathsf{vk}_Q)$ and also $\Sigma_Q.\mathsf{Verify}(\mathsf{vk}_Q, \sigma_Q, \mu)$, so to win an adversary either produced a forgery for $\Sigma$ or for some $\Sigma_Q$. The factor $Q'$ is an artefact of the reduction guessing which instance of $\Sigma_Q$ the adversary will produces a forgery on; there are at most $Q'$ many of these.

*Performance.* As discussed above, the appeal of the online-offline approach and the generic construction given in Figure 2 is to reduce latency when signing messages by pre-computing expensive steps. This comes at a cost of bandwidth (one verification key and two signatures are output by the signing algorithm) and at a cost for verification (two signatures need to be checked). We note that the bandwidth cost can be amortised in some settings.

### 2.4 GPV Signatures

The framework proposed by Gentry, Peikert, and Vaikuntanathan [GPV08], commonly known as the GPV framework, provides a generic and provably secure hash-and-sign framework based on trapdoor sampling and the Random Oracle Model, see Figure 3. It can be seen as a provable "fix" for the GGH signature scheme. The key ingredient here is a sampling algorithm that outputs signatures $\mathbf{s}$ whose distribution is statistically close to being independent of the trapdoor

$\mathsf{td}$. Let $(\mathsf{TrapGen}, \mathsf{SampD}, \mathsf{SampPre})$ be PPT algorithms with the following syntax and properties [GPV08,MP12,GM18]:

- $(\mathbf{A}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ takes dimensions $n, m \in \mathbb{N}$, a modulus $q \in \mathbb{N}$. It generates a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a trapdoor $\mathsf{td}$. For any $n \in \mathsf{poly}(\lambda)$ and $m \geq 5\, n \log q$, the distribution of $\mathbf{A}$ is within $\mathsf{negl}(\lambda)$ statistical distance to the uniform distribution on $\mathbb{Z}_q^{n \times m}$.
- $\mathbf{u} \leftarrow \mathsf{SampD}(1^n, 1^m, s)$ with $m \geq 5\, n \log q$ outputs an element in $\mathbf{u} \hookleftarrow\$\ D_{\mathbb{Z}^m, s}$. We have that $\mathbf{v} \coloneqq \mathbf{A} \cdot \mathbf{u} \bmod q$ is within $\mathsf{negl}(\lambda)$ statistical distance to the uniform distribution on $\mathbb{Z}_q^n$ if $s \geq \eta_\varepsilon(\Lambda_q^\perp(\mathbf{A}))$ for some $\varepsilon \in \mathsf{negl}(\lambda)$.
- $\mathbf{u}' \leftarrow \mathsf{SampPre}(\mathsf{td}, \mathbf{v}, s)$ with $m \geq 5\, n \log q$ takes a trapdoor $\mathsf{td}$, a vector $\mathbf{v} \in \mathbb{Z}_q^n$ and a parameter $s$. It samples $\mathbf{u}' \in \mathbb{Z}^m$ satisfying $\mathbf{A} \cdot \mathbf{u}' \equiv \mathbf{v} \bmod q$. Furthermore, $\mathbf{u}'$ is within $\mathsf{negl}(\lambda)$ statistical distance to $\mathbf{u} \leftarrow \mathsf{SampD}(1^n, 1^m, s)$ conditioned on $\mathbf{v} \equiv \mathbf{A} \cdot \mathbf{u} \bmod q$.

$\mathsf{KeyGen}(1^\lambda)$

$\mathbf{A}, \mathsf{td} \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$

**return** $\mathsf{vk} = \mathbf{A}, \mathsf{sk} = \mathsf{td}$

$\mathsf{Sign}(\mu, \mathsf{sk})$

$\mathbf{r} \hookleftarrow\$\ \{0,1\}^\lambda$; $\mathbf{y} \leftarrow \mathsf{SampPre}(\mathsf{td}, H(\mu, \mathbf{r}), s)$

**return** $(\mathbf{y}, \mathbf{r})$

$\mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$

**return** $\|\mathbf{y}\| \overset{?}{\leq} \beta' \wedge H(\mu, \mathbf{r}) \overset{?}{\equiv} \mathbf{A} \cdot \mathbf{y}$

**Fig. 3.** GPV Signatures

### 2.5 The Falcon Signature Scheme

Falcon [PFH+22] is a signature scheme based on the GPV framework and essentially makes two optimisations, these being the use of NTRU lattices [SS11] and the use of fast Fourier sampling [DP16]. Specialization to NTRU lattices necessitates assuming the hardness of lattice problems restricted to NTRU lattices. Since NTRU lattices are modules over cyclotomic rings, cryptanalysts have algebraic structure to try and exploit. A vulnerability resulting from the restriction to NTRU lattices is known in the "overstretched" regime, where the modulus $q$ is large in comparison to the dimension $n$. These regimes are well studied [ABD16,KF17,DvW21] and are to be avoided. Despite the apparent structure, lattice problems over NTRU lattices have persevered and there is reason to believe they remain hard [LS15]. Fast Fourier sampling on the other hand is an algorithmic speed up that does not affect the security assumptions. Recall NTRU lattice notation from Section 2. In particular, $n$ is a power of 2, $q$ is a prime number such that $q = 1 \bmod \varphi(n)$, and $\mathcal{O}_n \cong \mathbb{Z}(X)/\langle X^n + 1\rangle$ is the ring of integers of the 2n-th cyclotomic field $\mathbb{Q}(X)/\langle X^n + 1\rangle$. Falcon key generation,

signature generation, and signature verification are shown in Figure 4. We note that for consistency with [PFH+22], we present Falcon in Figure 4 such that vectors are row vectors instead of column vectors in contrast to the rest of this work. Throughout, we let $\odot$ denote multiplication in the Fourier domain which extends coordinate-wise to vectors and matrices.

*Key Generation.* The KeyGen procedure of Falcon consists of two main steps:

- **Solving an NTRU equation** involves generating polynomials for the secret-key $f$ and $g$ with small integer coefficients, as well as generating of $F, G \in \mathcal{O}_n$ such that $f \cdot G - g \cdot F = q$, using NTRUGen.
- **Trapdoor generation** is roughly equivalent to the TrapGen procedure in GPV. It uses the generated polynomials, $f, g, F, G \in \mathbb{Z}[X]/(X^n+1)$, and with some processing, makes them more compact. The matrix formed of these negacyclic coefficient blocks, $\mathbf{B} \leftarrow \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}^t$, is expressed in FFT domain as $\hat{B}$, and is then used to create $\hat{\mathbf{G}} \leftarrow \hat{\mathbf{B}}^* \cdot \hat{\mathbf{B}}$, which we then use to finally generate a compact trapdoor $\mathsf{T} \leftarrow \mathsf{ffLDL}^*(\hat{\mathbf{G}})$. Superscripts $\hat{}$ and $^*$ refer to FFT and complex conjugation, respectively. The procedure then returns $\mathsf{sk} := (\hat{\mathbf{B}}, \mathsf{T})$ and $\mathsf{vk} = h$, where $h \leftarrow g \cdot f^{-1} \mod q$.

*Signature Generation.* The Sign procedure of Falcon contains three main components:

- **Hashing the message** returns a random vector in $\mathbb{Z}_q^n$ (in the Random Oracle Model). Using SHAKE, some random salt and the message are hashed to produce a polynomial $c \in \mathbb{Z}_q[X]$ of degree $n$.
- **Trapdoor sampling** computes the signature without leaking the secret key. Falcon employs a fast Fourier sampler called ffSampling, which takes as input $\mathbf{t}$, which is a preimage of $c$, and $\mathsf{T}$, which is a Falcon tree apart of the secret key and is pre-processed in key generation. The resulting vector $\mathbf{z}$ is an element of the kernel lattice close to $\mathbf{t}$. The uncompressed signature becomes $\mathbf{s} = \hat{\mathbf{B}} \odot (\mathbf{t} - \mathbf{z})$.
- **Compression** takes the inverse FFT of $\mathbf{s}$, denoted $(s_0, s_1)$, satisfying $s_0 + s_1 \cdot h = c \mod q$ and returns $s_1$, encoded as a bitstring.

*Signature Verification.* The Verify procedure of Falcon mainly contains reconstructions and bound checks:

- **Reconstructions**. The target $c$ is computed from the message and salt, as is $s_1$ using the Decompress procedure. There also exists sufficient information to reconstruct $s_0 = c - s_1 \cdot h \mod q$.
- **Bound checks.** The reconstructed signature pair $(s_0, s_1)$ are finally accepted so long as they are small enough, i.e. $||(s_0, s_1)||^2 \leq \lfloor \beta^2 \rfloor$.

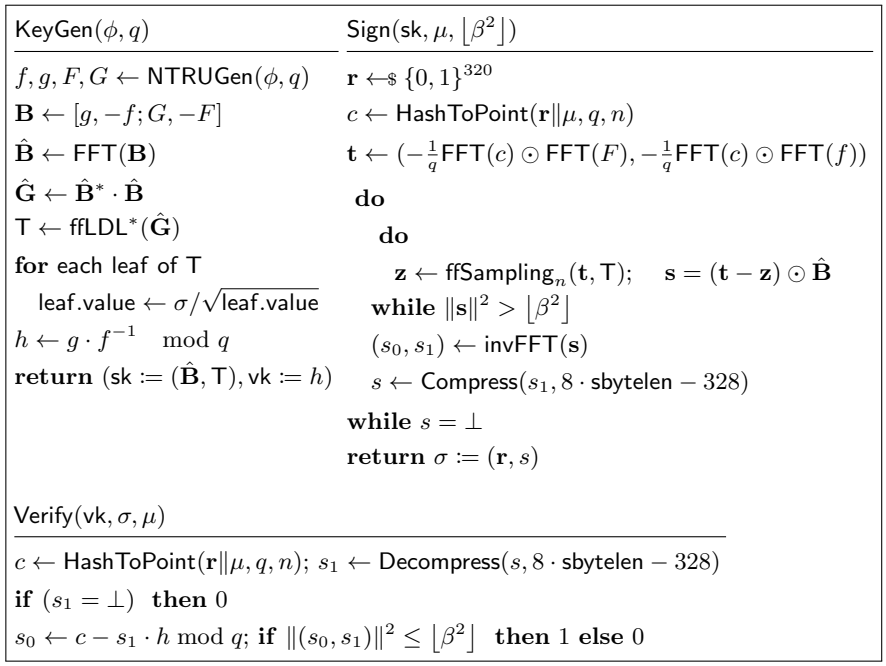


**Fig. 4.** The Falcon signature scheme [PFH+22].

## 3 GPV-like Few-Time Signatures

The online/offline paradigm relies on a one-time signature scheme for the online signing part. In previous iterations of the paradigm this has been achieved via a Merkle one-time signatures [EGM90,EGM96] or by utilising trapdoor hash functions [ST01].

In our construction, we base our one-time signature scheme on GGH signatures in "GPV-style", by which we mean instantiating GPV signatures without trapdoor sampling that ensures the outputs are independent of the trapdoor td. Since we are only using the trapdoor a few times, we can bound the leakage and thus security loss. We give our construction in Figure 5. The only difference is that we precompute $Q$ preimages $\mathbf{x}_{\mathrm{pre},i}$ in key generation, that we "mix in" these preimages during signing that we call $\mathsf{SampPre}_{\gg}$ instead of SampPre when signing.

Unlike SampPre, which is used in Falcon, we can be instead instantiate a more efficient procedure, $\mathsf{SampPre}_{\gg}$, that is indifferent to the output distribution, beyond that the outputs are short. This grants us the freedom to instantiate $\mathsf{SampPre}_{\gg}$ with fast pre-image generating algorithms, without any randomness requirements. We give an example in Figure 7 based on Babai rounding. By "mix in" we mean taking the sum of $x_{pre,c}$ and $x$ in Sign, which masks $x_{pre,c}$. We use the homomorphic properties on the images and pre-images to assure the mixing works.

Two natural candidates for $\mathsf{SampPre}_{\gg}$ are Babai's rounding and Nearest plane algorithms. The latter, while slower should allow for more online signatures without refreshing the key. We chose the former in our implementation for its speed, facilitating for rapid online signatures.

$\mathsf{KeyGen}(1^\lambda, Q)$

$\mathbf{A}, \mathsf{td} \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$; $c \leftarrow 0$
**for** $0 \le i < Q$: $\mathbf{x}_{\mathrm{pre},i} \leftarrow \mathsf{SampD}(1^n, 1^m, s)$
**return** $\mathsf{vk} = \mathbf{A}, \mathsf{sk} = (\mathsf{td}, \{\mathbf{x}_{\mathrm{pre},i}\}_{0 \le i < Q}, c)$

$\mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$

**return** $\|\mathbf{y}\| \overset{?}{\le} \beta' \wedge H(\mu, \mathbf{r}) \overset{?}{\equiv} \mathbf{A} \cdot \mathbf{y}$

$\mathsf{Sign}(\mu, \mathsf{sk})$

$\mathbf{r} \leftarrow\!\!\$\ \{0,1\}^\lambda$
$\mathbf{t}' \leftarrow \mathbf{A} \cdot \mathsf{sk}.\mathbf{x}_{\mathrm{pre},c}$; $\mathbf{t} \leftarrow H(\mu, \mathbf{r})$
$\mathbf{x} \leftarrow \mathsf{SampPre}_{\gg}(\mathsf{sk}.\mathsf{td}, \mathbf{t} - \mathbf{t}')$
$\mathbf{y} \leftarrow \mathbf{x}_{\mathrm{pre},c} + \mathbf{x}$
$\mathsf{sk}.c \leftarrow \mathsf{sk}.c + 1$
**return** $(\mathbf{y}, \mathbf{r})$

**Fig. 5.** Few-Times GPV-style Signatures

### 3.1 Unforgeability in the ROM

We prove the following theorem to establish the EUF-CMA$_Q$ security of our few time signatures construction in the ROM.

**Theorem 1 (Unforgeability).** *The advantage* $\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma,Q}(\lambda)$ *of any* $Q$*-query bounded adversary* $\mathcal{A}$ *playing the* EUF-CMA$_Q$ *security in the ROM game pictured in Figure 6 successfully forges is bounded as*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma,Q}(\lambda) \le Q \cdot \mathsf{negl}(\lambda) \cdot 2^{Q \cdot \left(\log_2\left(1 + \frac{2\pi\|\mathbf{x}\|^2_{\max}}{s^2}\right) + \frac{\log_2 e}{2}\right)} \cdot \mathsf{Adv}^{\text{sis}}_{\sqrt{2\,m}\cdot s}(\lambda)$$

*when* $\frac{s^2_{\mathsf{sim}}}{2\pi} = \frac{s^2}{2\pi} + \mathbb{E}_{\mathbf{x}}\left(\|\mathbf{x}\|^2\right)$ *and*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma,Q}(\lambda) \le Q \cdot \mathsf{negl}(\lambda) \cdot 2^{Q \cdot \left(\log_2\left(1 + \frac{2\pi\mathbb{E}_{\mathbf{x}}\left(\|\mathbf{x}\|^2\right)}{s^2}\right) + \frac{\|\mathbf{x}\|^2_{\max}\log_2 e}{2\mathbb{E}_{\mathbf{x}}\left(\|\mathbf{x}\|^2\right)}\right)} \cdot \mathsf{Adv}^{\text{sis}}_{\sqrt{2\,m}\cdot s}(\lambda)$$

*when* $\frac{s^2_{\mathsf{sim}}}{2\pi} := \frac{s^2}{2\pi} + \|\mathbf{x}\|^2_{\max}$ *and where,*

- $\mathsf{Adv}^{\text{sis}}_{\sqrt{2\,m}\cdot s}(\lambda)$ *is advantage of any PPT adversary against the Short Integer Solutions problem with norm bound* $\sqrt{2\,m} \cdot s$.
- $s$ *is the parameter of the spherical discrete Gaussian* $\mathbf{x}_{pre}$,
- $s_{\mathsf{sim}}$ *is the parameter of a simulated spherical discrete Gaussian, and*
- $\|\mathbf{x}\|^2_{\max}$ *is the maximum* $\|\mathbf{x}\|^2$ *over* $\mathbf{x} \sim \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathcal{U}(\mathbb{Z}_q^n))$, *and*
- $\lambda$ *is the security parameter.*

| $\mathrm{EUF\text{-}CMA}^{\mathcal{A}}_{\Sigma,Q}(\lambda)$ | $\mathsf{SIGN}(\mu)$ | $\mathsf{RO}(x)$ |
|---|---|---|
| $\mathcal{Q}, c \leftarrow \emptyset, 0$ | **if** $c \geq Q$ **then abort** | **if** $x \notin \mathcal{H}$ **then** |
| $\mathsf{vk}, \mathsf{sk} \leftarrow \Sigma.\mathsf{KeyGen}(1^\lambda)$ | $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \rho_c, \mu)$ | $\mathcal{H}[x] \leftarrow\!\!\$\ \mathbb{Z}_q^n$ |
| $\{\rho_i \leftarrow \Sigma.\mathsf{PreSign}(\mathsf{sk})\}_{1 \leq i \leq Q}$ | $c \leftarrow c + 1$ | **return** $\mathcal{H}[x]$ |
| $(\mu^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathsf{SIGN}}(\mathsf{vk})$ | $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$ | |
| **return** $(\mu^\star, \cdot) \notin \mathcal{Q} \wedge \mathsf{Verify}(\mathsf{vk}, \sigma^\star, \mu^\star)$ | **return** $\sigma$ | |

**Fig. 6.** $\mathrm{EUF\text{-}CMA}_Q$ in the ROM.

We prove Theorem 1 using a series of game hops in Appendix A, following the standard GPV proof pattern. We next outline the hops, deferring the details to Appendix A. $\mathsf{Game}_1$ programs the random oracle to answer hash queries as the syndrome of a discrete Gaussian sampler with a possibly larger parameter. $\mathsf{Game}_2$ ensures message-salt pairs are not repeated. $\mathsf{Game}_3$ simulates signatures as pre-images of targets obtained from calling the random oracle on salted messages. Justifying the hop from $\mathsf{Game}_2$ to $\mathsf{Game}_3$ is the non-trivial part of the proof, captured by the following lemma 1. $\mathsf{Game}_4$ forgets the trapdoor basis matrix and in its place draws one at random. The proof is completed by transforming an adversary that wins $\mathsf{Game}_4$ into an SIS solver.

**Lemma 1 (Query bounded advantage playing $\mathsf{Game}_3$ over $\mathsf{Game}_2$).** *For every adversary $\mathcal{A}_Q$ playing $\mathsf{Game}_2$ or $\mathsf{Game}_3$, making at most $Q$ signature queries before presenting the forgery, and every $a \in (1, \infty)$,*

$$\frac{\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_2}(\mathsf{vk}) = 1\right]}{\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_3}(\mathsf{vk}) = 1\right]^{\frac{a-1}{a}}} \leq \left(\frac{s_{\mathsf{sim}}^2}{s^2}\right)^Q \cdot (1+\epsilon)^Q \cdot \exp\left(\frac{(a-1) \cdot \pi \cdot \|\mathbf{x}\|_{\max}^2 Q}{\left(a \cdot (\frac{s_{\mathsf{sim}}^2}{s^2} - 1) + 1\right) \cdot s^2}\right) \tag{4}$$

*where,*

- *$s$ is the parameter of the spherical discrete Gaussian in $\mathsf{Game}_2$ $\mathbf{x}_{pre}$,*
- *$s_{\mathsf{sim}}$ is the parameter of the simulator spherical discrete Gaussian in $\mathsf{Game}_3$,*
- *$\|\mathbf{x}\|_{\max}^2$ is the maximum $\|\mathbf{x}\|^2$ over $\mathbf{x} \sim \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathcal{U}(\mathbb{Z}_q^n))$.*
- *and $\epsilon$ is a negligible function in the security parameter with $s_{\mathsf{sim}} \geq s$ and $s_{\mathsf{sim}} \geq \sqrt{\ln(2m(1+1/\epsilon)/\pi)} \cdot \sqrt{(a((s_{\mathsf{sim}}^2/s^2) - 1) + 1)}$.*

*Proof.* Consider an adversary $\mathcal{A}_Q$ playing $\mathsf{Game}_2$ (Figure 9) or $\mathsf{Game}_3$ (Figure 10). Without loss of generality, assume that the adversary $\mathcal{A}_Q$ makes exactly $Q$ signature queries before presenting the forgery. Let $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_Q)$ denote the signatures seen by $\mathcal{A}_Q$ while playing Game 2, that is, if the queries were answered by the signing algorithm. Let $\mathcal{D}_1$ denote the join distribution of such a sequence $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_Q)$. Let $(\mathbf{y}_1^{\mathsf{sim}}, \mathbf{y}_2^{\mathsf{sim}}, \ldots, \mathbf{y}_Q^{\mathsf{sim}})$ denote the sequence of signature query answers seen by $\mathcal{A}_Q$ while playing Game 3, that is, if the simulator $\mathsf{Sim}$ answers the signature queries. Let $\mathcal{D}_2$ denote the joint distribution of such a sequence

$(\mathbf{y}_1^{\mathsf{sim}}, \mathbf{y}_2^{\mathsf{sim}}, \ldots, \mathbf{y}_Q^{\mathsf{sim}})$. Let $E$ be the event of a successful adversary forgery at the end. Then $\mathcal{D}_1(E) = \Pr\left[\mathcal{A}_Q^{\mathrm{Game}_2}(\mathsf{vk}) = 1\right]$ and $\mathcal{D}_2(E) = \Pr\left[\mathcal{A}_Q^{\mathrm{Game}_3}(\mathsf{vk}) = 1\right]$ are the probability that the adversary succeeds in Game 2 and Game 3 respectively. From the probability preservation of Rényi entropy (Equation (2)), for $a \in (1, \infty)$

$$\mathcal{D}_1(E) \leq \left[\mathcal{D}_2(E) R_a\left(\mathcal{D}_1 || \mathcal{D}_2\right)\right]^{\frac{a-1}{a}}. \tag{5}$$

Therefore, to claim the bound in the lemma, it suffices to bound $R_a(\mathcal{D}_1 \| \mathcal{D}_2)$. We begin by decomposing $\mathcal{D}_1$ and $\mathcal{D}_2$ into distributions indexed by the signature queries. Regardless of the if Game 2 or Game 3 is played, the signatures generated are mutually independent across queries. Therefore, both distributions factor into products $\mathcal{D}_1 = \prod_{j=1}^{Q} \mathcal{D}_{1,j}$ and $\mathcal{D}_2 = \prod_{j=1}^{Q} \mathcal{D}_{2,j}$, where $\mathcal{D}_{1,j}$ and $\mathcal{D}_{2,j}$ are respectively the distributions followed by $\mathbf{y}_j$ and $\mathbf{y}_j^{\mathsf{sim}}$. By the multiplicativity of Rényi divergence, for $a \in [1, \infty]$,

$$R_a\left(\prod_{j=1}^{Q} \mathcal{D}_{1,j} \middle\| \prod_{j=1}^{Q} \mathcal{D}_{2,j}\right) = \prod_{j=1}^{Q} R_a(\mathcal{D}_{1,j} \| \mathcal{D}_{2,j}). \tag{6}$$

Further, $\mathcal{D}_{1,j}, 1 \leq j \leq Q$ are all identical, whose distribution we next determine. Since the signature vector $\mathbf{y} \leftarrow \mathbf{x} + \mathbf{x}_{\mathrm{pre}}$ in $\mathsf{Game}_2$ is the sum of independent vectors $\mathbf{x}$ and $\mathbf{x}_{\mathrm{pre}}$, its distribution is the convolution

$$\mathcal{D}_{\mathbb{Z}^m, s} \circ \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathcal{U}(\mathbb{Z}_q^n)),$$

of the distributions of $\mathbf{x}$ and $\mathbf{x}_{\mathrm{pre}}$. Written explicitly, the signature $\mathbf{y}$ follows the distribution $\mathbf{w} \in \mathbb{Z}^m \longmapsto \Pr[\mathbf{y} = \mathbf{w}]$ mapping

$$\mathbf{w} \longmapsto \frac{\mathbb{E}_{\mathbf{x}}\left[\exp(-\pi ||\mathbf{w} - \mathbf{x}||^2 / s^2)\right]}{\sum_{\mathbf{w} \in \mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \exp(-\pi ||\mathbf{w} - \mathbf{x}||^2 / s^2)},$$

where the expectation is taken over the pre-images $\mathbf{x} \sim \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathcal{U}(\mathbb{Z}_q^n))$ induced by uniform images $\mathcal{U}(\mathbb{Z}_q^n)$. We denote this distribution as $\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m, s, -\mathbf{x}}\right)$. In particular, $\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m, s, -\mathbf{x}}\right)$ has mean zero vector and covariance $s^2 \cdot \mathbb{I}_n + \mathrm{Var}(\mathbf{x})$.

Likewise, $\mathcal{D}_{1,j}, 1 \leq j \leq Q$ are all identical, being $\mathcal{D}_{\mathbb{Z}^m, s_{\mathsf{sim}}}$ by construction. Substituting for $\mathcal{D}_{1,j}$ and $\mathcal{D}_{2,j}$ in Equation (6), for $a \in [1, \infty]$,

$$R_a\left(\prod_{j=1}^{Q} \mathcal{D}_{1,j} \middle\| \prod_{j=1}^{Q} \mathcal{D}_{2,j}\right) = R_a(\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m, s, -\mathbf{x}}\right) \| \mathcal{D}_{\mathbb{Z}^m, s_{\mathsf{sim}}})^Q. \tag{7}$$

Therefore, it suffices to upper bound $R_a(\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m, s, -\mathbf{x}}\right) \| \mathcal{D}_{\mathbb{Z}^m, s_{\mathsf{sim}}})$. From the definition of Rényi divergence (Equation (1)),

$$\begin{aligned} &R_a(\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m, s, -\mathbf{x}}\right) \| \mathcal{D}_{\mathbb{Z}^m, s_{\mathsf{sim}}})^{a-1} \\ &= \frac{\left[\sum_{\mathbf{w} \in \mathbb{Z}^m} \exp(-\pi \|\mathbf{w}\|^2 / s_{\mathsf{sim}}^2)\right]^{a-1}}{\left[\sum_{\mathbf{w} \in \mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \exp(-\pi \|\mathbf{w} - \mathbf{x}\|^2 / s^2)\right]^a} \sum_{\mathbf{w} \in \mathbb{Z}^m} \frac{\left[\mathbb{E}_{\mathbf{x}} \exp(-\pi \|\mathbf{w} - \mathbf{x}\|^2 / s^2)\right]^a}{\exp(-\pi (a-1) \|\mathbf{w}\|^2 / s_{\mathsf{sim}}^2)}. \end{aligned}$$

For each $\mathbf{w} \in \mathbb{Z}^m$, by Jensen's inequality

$$\left[\mathbb{E}_{\mathbf{x}} \exp(-\pi \|\mathbf{w}-\mathbf{x}\|^2/s^2)\right]^a \leq \mathbb{E}_{\mathbf{x}} \exp(-a\cdot\pi \|\mathbf{w}-\mathbf{x}\|^2/s^2),$$

since raising the positive random variable $\exp(-\pi \cdot \|\mathbf{w}-\mathbf{x}\|^2/s^2)$ to the $a$-th power is convex for $a \geq 1$. Therefore,

$$\begin{aligned}
&R_a(\mathbb{E}_{\mathbf{x}}\,(\mathcal{D}_{\mathbb{Z}^m,s,-\mathbf{x}})\,\|\mathcal{D}_{\mathbb{Z}^m,s_{\mathsf{sim}}})^{a-1} \\
&\quad\leq \frac{\left[\sum_{\mathbf{w}\in\mathbb{Z}^m} \exp(-\pi\cdot\|\mathbf{w}\|^2/s_{\mathsf{sim}}^2)\right]^{a-1}}{\left[\sum_{\mathbf{w}\in\mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \exp(-\pi\cdot\|\mathbf{w}-\mathbf{x}\|^2/s^2)\right]^a} \sum_{\mathbf{w}\in\mathbb{Z}^m} \frac{\mathbb{E}_{\mathbf{x}} \exp(-a\cdot\pi\cdot\|\mathbf{w}-\mathbf{x}\|^2/s^2)}{\exp(-\pi\cdot(a-1)\cdot\|\mathbf{w}\|^2/s_{\mathsf{sim}}^2)}.
\end{aligned}$$

First, let us estimate the normalisation factor. Since $\mathbb{Z}^m$ has $m$ $\mathbb{R}-$linearly independent vectors of length one, by [MR04][Lemma 3.3] the smoothing parameter $\eta_\epsilon(\mathbb{Z}^m) \leq \sqrt{\ln(2m(1+1/\epsilon)/\pi)}$ and for $s_{\mathsf{sim}} \geq \sqrt{\ln(2m(1+1/\epsilon)/\pi)}$ we have,

$$\sum_{\mathbf{w}\in\mathbb{Z}^m} \exp(-\pi\cdot\|\mathbf{w}\|^2/s_{\mathsf{sim}}^2) \leq s_{\mathsf{sim}}(1+\epsilon).$$

By the linearity of expectation:[1]

$$\sum_{\mathbf{w}\in\mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \exp(-\pi\cdot\|\mathbf{w}-\mathbf{x}\|^2/s^2) = \mathbb{E}_{\mathbf{x}} \sum_{\mathbf{w}\in\mathbb{Z}^m} \exp(-\pi\cdot\|\mathbf{w}-\mathbf{x}\|^2/s^2) \geq s.$$

Therefore,

$$R_a(\mathbb{E}_{\mathbf{x}}\,(\mathcal{D}_{\mathbb{Z}^m,s,-\mathbf{x}})\,\|\mathcal{D}_{\mathbb{Z}^m,s_{\mathsf{sim}}})^{a-1} \leq \frac{s_{\mathsf{sim}}^{a-1}(1+\epsilon)^{a-1}}{s^a} \sum_{\mathbf{w}\in\mathbb{Z}^m} \frac{\left[\mathbb{E}_{\mathbf{x}} \exp(-a\cdot\pi\cdot\|\mathbf{w}-\mathbf{x}\|^2/s^2)\right]}{\exp(-\pi\cdot(a-1)\cdot\|\mathbf{w}\|^2/s_{\mathsf{sim}}^2)}.$$

By the linearity of expectation, the right hand side equals

$$\begin{aligned}
&\frac{s_{\mathsf{sim}}^{a-1}(1+\epsilon)^{a-1}}{s^a} \sum_{\mathbf{w}\in\mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \left[\frac{\exp(-a\cdot\pi\cdot\|\mathbf{w}-\mathbf{x}\|^2/s^2)}{\exp(-\pi\cdot(a-1)\cdot\|\mathbf{w}\|^2/s_{\mathsf{sim}}^2)}\right] \\
&= \frac{s_{\mathsf{sim}}^{a-1}(1+\epsilon)^{a-1}}{s^a} \sum_{\mathbf{w}\in\mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \exp\left(-\pi\cdot\left(\frac{a\cdot\|\mathbf{w}-\mathbf{x}\|^2}{s^2} - \frac{(a-1)\cdot\|\mathbf{w}\|^2}{s_{\mathsf{sim}}^2}\right)\right).
\end{aligned}$$

Setting $r := s_{\mathsf{sim}}^2/s^2$ and completing the squares in the exponent, we get,

$$\begin{aligned}
\frac{r^{a-1}(1+\epsilon)^{a-1}}{s} \sum_{\mathbf{w}\in\mathbb{Z}^m} \mathbb{E}_{\mathbf{x}} \exp\Bigg(\frac{-\pi}{s_{\mathsf{sim}}^2}\cdot\Bigg\{&(a\cdot(r-1)+1)\cdot\|\mathbf{w}-\frac{a\cdot r}{a\cdot(r-1)+1}\mathbf{x}\|^2 \\
&- a\cdot r\left(\frac{a\cdot r}{a\cdot(r-1)+1}-1\right)\|\mathbf{x}\|^2\Bigg\}\Bigg),
\end{aligned}$$

[1] There are no convergence issues here, or later in the proof, when we exchange infinite summations by claiming linearity of expectation, as the summands are not negative.

which by the linearity of expectation,

$$= \frac{r^{a-1}(1+\epsilon)^{a-1}}{s} \cdot \mathbb{E}_{\mathbf{x}}\Bigg[ \exp\left(\frac{\pi \cdot a}{s^2}\left(\frac{a \cdot r}{a\cdot(r-1)+1} - 1\right)\|\mathbf{x}\|^2\right) \cdot \sum_{\mathbf{w}\in\mathbb{Z}^m} \exp\left(\frac{-\pi(a(r-1)+1)}{s_{\mathsf{sim}}^2}\|\mathbf{w} - \frac{a\cdot r}{a\cdot(r-1)+1}\mathbf{x}\|^2\right)\Bigg].$$

As before, by [MR04][Lemma 3.3],

$$\sum_{\mathbf{w}\in\mathbb{Z}^m} \exp\left(\frac{-\pi(a(r-1)+1)}{s_{\mathsf{sim}}^2}\|\mathbf{w} - \frac{ar}{a(r-1)+1}\mathbf{x}\|^2\right) \leq \frac{s_{\mathsf{sim}}(1+\epsilon)}{\sqrt{(a(r-1)+1)}},$$

since $s_{\mathsf{sim}} \geq \sqrt{\ln(2m(1+1/\epsilon)/\pi)} \cdot \sqrt{(a((s_{\mathsf{sim}}^2/s^2)-1)+1)}$. In summary,

$$\begin{aligned}
&R_a(\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m,s,-\mathbf{x}}\right)\|\mathcal{D}_{\mathbb{Z}^m,s_{\mathsf{sim}}})^{a-1} \\
&\leq \frac{r^a(1+\epsilon)^a}{\sqrt{(a\cdot(r-1)+1)}}\mathbb{E}_{\mathbf{x}}\left[\exp\left(\frac{\pi a}{s^2}\left(\frac{a-1}{a\cdot(r-1)+1}\right)\|\mathbf{x}\|^2\right)\right] \\
&\leq \frac{r^a(1+\epsilon)^a}{\sqrt{(a\cdot(r-1)+1)}} \cdot \exp\left(\frac{\pi\cdot a}{s^2}\left(\frac{a-1}{a(r-1)+1}\right)\|\mathbf{x}\|_{max}^2\right).
\end{aligned}$$

Therefore,

$$\begin{aligned}
&R_a(\mathbb{E}_{\mathbf{x}}\left(\mathcal{D}_{\mathbb{Z}^m,s,-\mathbf{x}}\right)\|\mathcal{D}_{\mathbb{Z}^m,s_{\mathsf{sim}}})^{\frac{a-1}{a}} \\
&\leq \left(\frac{r^a(1+\epsilon)^a}{\sqrt{(a\cdot(r-1)+1)}}\right)^{\frac{1}{a}} \cdot \exp\left(\frac{\pi\beta}{s^2}\left(\frac{a-1}{a\cdot(r-1)+1}\right)\cdot\|\mathbf{x}\|_{\max}^2\right) \\
&\leq r\cdot(1+\epsilon)\cdot\exp\left(\frac{\pi\cdot(a-1)\cdot\|\mathbf{x}\|_{\max}^2}{(a\cdot(r-1)+1)\cdot s^2}\right).
\end{aligned}$$

By the probability preservation (Equation (5)) and multiplicativity (Equation (7)) of Rényi entropy,

$$\frac{\Pr\left[\mathcal{A}_Q^{Game_2}(\mathsf{vk})=1\right]}{\Pr\left[\mathcal{A}_Q^{Game_3}(\mathsf{vk})=1\right]^{\frac{a-1}{a}}} \leq \left(\frac{s_{\mathsf{sim}}^2}{s^2}\right)^Q \cdot (1+\epsilon)^Q \cdot \exp\left(\frac{(a-1)\cdot\pi\cdot\|\mathbf{x}\|_{\max}^2 Q}{\left(a\cdot(\frac{s_{\mathsf{sim}}^2}{s^2}-1)+1\right)\cdot s^2}\right) \tag{8}$$

which is exactly as claimed in the lemma. □

We restrict the Renyi divergence parameter $a$ in the lemma to $a > 1$, allowing calculations with $a-1$ in denominators. While applying the lemma in security arguments, we may optimise over $a$. Therefore, excluding $a = 1$ is not an issue.

The constraint $s_{\mathsf{sim}} \geq \sqrt{\ln(2m(1+1/\epsilon)/\pi)} \cdot \sqrt{(a((s_{\mathsf{sim}}^2/s^2)-1)+1)}$ in the lemma appears too restrictive to apply for large $a$. This is pessimistic and an

artifact of our proof where we used the same $\epsilon$ in both applications of the smoothing parameter argument [MR04][Lemma 3.3]. The second application only contributes a multiplicative factor $(1+\epsilon)^{Q/a}$ to the bound. A more refined analysis with two $\epsilon$ parameters is possible, which we exclude for ease of exposition. For many choices of $a$, the constraint $s_{\mathsf{sim}} \geq \sqrt{\ln(2m(1+1/\epsilon)/\pi)}$ might suffice.

*Remark 1.* As a sanity check, we may consider the parallelepiped attack [NR06] on NTRU signatures (and its extensions [DN12]). These attack may try to learn the ephemeral trapdoor basis from the online signatures. Lemma 1 implies that (for well chosen parameters) from only a few online signatures $Q$ this will fail. As the number of signatures $Q$ increases, our bound becomes meaningless, but this does not immediately imply an attack, i.e. it is not a priori clear our bound is tight. We leave to future work to establish tighter bounds, allowing potentially larger $Q$.

## 4 A Falcon-based Online-Offline Signature Scheme

In this section we specialise and analyse the few-times signatures from GPV developed in Section 3 to the Falcon-512 setting, by restricting to the NTRU lattice underlying Falcon-512 with lattice dimension $m = 2 \cdot 512$, image space dimension $n = 512$ and modulus $q = 12,289$. Our Falcon-inspired few-times signature scheme is presented in Figure 7, again following the notation in Falcon (as described in Section 2.5), as opposed to GPV.

*Remark 2.* Many of the elements in Figure 7 are similar or the same as in Falcon, with a few exceptions. $\mathsf{PreSign}$ offloads some of the FFT conversions on the secret key needed during Falcon's signature generation procedure. In this part we also add the offline generation of two Gaussian vectors. $\mathsf{ComputeTarget}$ reuses operations required in Falcon's verification procedure. $\mathsf{Sign}$ reuses the initial salting and hashing from Falcon, as well as the final compression. However the remaining operations are bespoke, including a faster algorithm for sampling pre-images, $\mathsf{SampPre}_{\gg}$, and the algebra used to mask these pre-images. In Figure 7, $\mathsf{SampPre}_{\gg}$ is instantiated with the Babai method, using rounding. $\mathsf{PreSign}$ is computed offline, with $\mathsf{ComputeTarget}$, and the remaining operations are computed online.

In our implementation, the short pre-image function $\mathsf{SampPre}_{\gg}$ is implemented using Babai rounding. We observed empirically that pre-image lengths have expectation $\mathbb{E}_{\mathbf{x}}\|\mathbf{x}\|_2 = 5584.61$ and standard deviation $= 303.67$. Further, the pre-image lengths appear to be sub-Gaussian in practice, suggesting that except for a probability of 0.03%, the pre-image lengths do not exceed length $5584.61 + 3 \cdot 303.67 = 6495.62$. Guided by this empirical observation, we assume in the analysis that the maximum allowed pre-image length $\|\mathbf{x}\|_{\max} := 6495.62$. In one desires the analysis to strictly conform to the implementation, it is easy to add a rejection sampling step to $\mathsf{SampPre}_{\gg}$, rejecting pre-images larger than $\|\mathbf{x}\|_{\max}$.

$\mathsf{PreSign}(\mathsf{sk})$

$\mathbf{B} \leftarrow [g, -f; G, -F]$
$\hat{\mathbf{B}} \leftarrow \mathsf{FFT}(\mathbf{B})$
$u_0, u_1 \leftarrow \mathcal{D}_\sigma^2$
$c_{\text{pre}} \leftarrow \mathsf{ComputeTarget}(h, u_0, u_1)$
**return** $(\rho \coloneqq u_0, u_1, c_{\text{pre}})$

$\mathsf{Sign}(\mathsf{sk}, \rho, \mu)$

$\mathbf{r} \leftarrow\!\!\$\ \{0,1\}^{320}$
$c' \leftarrow \mathsf{HashToPoint}(\mathbf{r} \| \mu, q, n)$
$c \leftarrow c' + c_{\text{pre}}$
$s_0', s_1' \leftarrow \mathsf{SampPre}_{\gg}(c, \hat{\mathbf{B}})$
$s_0 \leftarrow s_0' - u_0$
$s_1 \leftarrow s_1' - u_1$
$s \leftarrow \mathsf{Compress}(s_1)$
**return** $\sigma \coloneqq (\mathbf{r}, s)$

$\mathsf{ComputeTarget}(h, u_0, u_1)$

$\hat{u_1} \leftarrow \mathsf{NTT}(u_1)$
$\hat{t} \leftarrow h \odot \hat{u_1}$
$t \leftarrow \mathsf{invNTT}(\hat{t}) + u_0$
**return** $t$

$\mathsf{SampPre}_{\gg}(c, \hat{\mathbf{B}})$

$\hat{c} \leftarrow \mathsf{FFT}(c)$
// $\hat{\mathbf{t}} = (\mathsf{FFT}(c), \mathsf{FFT}(0)) \cdot \hat{\mathbf{B}}^{-1}$
$\hat{\mathbf{t}} \leftarrow \left(-\frac{1}{q} \cdot \hat{c} \odot \mathsf{FFT}(F), -\frac{1}{q} \cdot \hat{c} \odot \mathsf{FFT}(f)\right)$
$\mathbf{t} \leftarrow \mathsf{invFFT}(\hat{\mathbf{t}})$; $\mathbf{t} \leftarrow \mathsf{round}(\mathbf{t})$; $\hat{\mathbf{t}} \leftarrow \mathsf{FFT}(\mathbf{t})$
$\hat{\mathbf{s}} \leftarrow (\mathsf{FFT}(c), \mathsf{FFT}(0)) - \hat{\mathbf{t}} \odot \hat{\mathbf{B}}$
$\mathbf{s} \leftarrow \mathsf{invFFT}(\hat{\mathbf{s}})$
**return** $\mathbf{s}$

**Fig. 7.** Few-Times Falcon-style signature generation.

We note that GPV security proof does not apply to Falcon, because it assumes $n = \Theta(\lambda)$, $q = \lambda^{\mathcal{O}(1)}$, and $m = \Theta(n \log q)$ whereas Falcon has $m = \Theta(n)$. Thus, like Falcon our scheme has no formal proof of security that reduces some known problem to forgeries. As is standard, we establish parameters using cost estimates for known cryptanalytic attacks. In our case, we rely on the "lattice estimator" from [APS15].

By matching the simulator and signature variance, we tabulate the security guarantee from Equation (11). The column "Zero" signature bit security is the bit security assured by the lattice estimator associated with solving SIS with length $\sqrt{(m \cdot s_{\mathsf{sim}}^2)/(2\pi)}$. The choice of parameters we implemented is highlighted in the table as "Implemented".

### 4.1 Practical Performance

Our implementation is publicly available[2] and reuses parts of the publicly available implementation of Falcon [Por19]. Our main changes, as shown in Figure 7, are the way in which the target is computed and the calculation of the short pre-image are reworked. At the same time, many of the typical modules, such

[2] https://github.com/jameshoweee/online-offline-sigs/

**Table 1.** Falcon-inspired few-times signature scheme security levels, with security level deterioration by key reuse, for a variety of pre-image and noise parameters.

| Pre-image length, Noise std. dev.* | Simulator length† | "Zero" sig bit security‡ | Bit security deficit per sig§ | 4 sig bit security | 8 sig bit security |
|---|---|---|---|---|---|
| Small noise 1 $(5\,584.61, 16)$ | 5 608.03 | 148.3 | 17.87 | 76.82 | 5.34 |
| Small noise 2 $(5\,584.61, 32)$ | 5 677.71 | 147.7 | 15.87 | 84.22 | 20.74 |
| Small noise 3 $(5\,584.61, 64)$ | 5 948.29 | 145.8 | 13.87 | 90.32 | 34.84 |
| Small noise 4 $(5\,584.61, 128)$ | 6 925.68 | 139.7 | 11.87 | 92.22 | 44.74 |
| Matching lengths¶ $(5\,584.61, 174.51)$ | 7 897.62 | 135.1 | 10.97 | 91.22 | 47.34 |
| Implemented‖ $(5\,584.61, 215)$ | 8 861.28 | 130.9 | 10.38 | 89.38 | 47.86 |
| Small noise 4 $(5\,584.61, 256)$ | 9 914.47 | 127.4 | 9.87 | 87.92 | 48.44 |
| Medium noise# $(5\,584.61, 987.23)$ | 32 081.14 | 98.0 | 6.02 | 73.94 | 49.84 |
| High noise** $(5\,584.61, 5\,584.61)$ | 178 794.00 | 71.8 | 1.98 | 63.89 | 55.99 |

* Describing the (length, noise) as $(\mathbb{E}_{\mathbf{x}}\|\mathbf{x}\|, \frac{s}{\sqrt{2\pi}})$,
† Using $\sqrt{\frac{ms_{\mathsf{sim}}^2}{2\pi}} := \sqrt{\frac{ms^2}{2\pi} + (\mathbb{E}_{\mathbf{x}}\|\mathbf{x}\|)^2}$
‡ Using the lattice estimator with length $\sqrt{(ms_{\mathsf{sim}}^2)/(2\pi)}$,
§ Using Equation (11)
‖ The parameters used in the implementation in Section 4.1,
¶ Where $\mathbb{E}_{\mathbf{x}}\|\mathbf{x}\| = \frac{s\sqrt{m}}{\sqrt{2\pi}}$
# Where $\mathbb{E}_{\mathbf{x}}\|\mathbf{x}\| = \frac{sm^{1/4}}{\sqrt{2\pi}}$,
** Where $\mathbb{E}_{\mathbf{x}}\|\mathbf{x}\| = \frac{s}{\sqrt{2\pi}}$

as the FFT multiplications, hash functions, and pseudorandomness generation remain as in Falcon. We also adopt the same parameters from the latest version of Falcon except for those which we use for discrete Gaussian sampling.

The discrete Gaussian sampler we require has significantly larger standard deviation than those used in Falcon. In Table 1 we describe security levels for different standard deviation values, which also defines the value we use in our proof-of-concept implementation, $\sigma = 215$. Since the standard deviation we use is larger, adapting the same naïve table-based sampler used in Falcon would be too slow, consume too much memory, and overall be too inefficient. Previous research [GLP12,BBE+19] dealt with this issue by adopting Peikert's convolution lemma [Pei10], which meant two Gaussian samplers from much smaller standard deviations could be combined to produce one Gaussian sample from a much larger distribution. This technique was also used in GALACTICS [BBE+19], which provides a secure, constant-time design for their Gaussian sampler. Overall, this ensures the clock cycle consumption remains low, which is important despite this being apart of the offline phase.

We give benchmarks in Table 3. All these benchmarks were run using a Raspberry Pi 3 Model B, Revision 1.2, equipped with an ARM Cortex-A53 CPU. The system is configured for AArch32 (32-bit), using the latest version of the Raspberry Pi OS (Bullseye), which is based on Debian 11. We employed

**Table 2.** Key and signature sizes (in bytes) of our Falcon-based online-offline signature scheme alongside Dilithium and Ed25519.

| Signature Scheme | Pubic Key | Secret Key | Signature |
|---|---|---|---|
| Falcon-512 | 897 | 7,553 | 666 |
| This work, uncompressed | 897 | $15,106 + \tau \cdot 1,024$ | 897+666+1,024 |
| This work, orig. compressed | 897 | $15,106 + \tau \cdot 845$ | 897+666+845 |
| This work, alt. compressed | 897 | $15,106 + \tau \cdot 780$ | 897+666+780 |
| Dilithium2 | 1,312 | 2,528 | 2,420 |
| Ed25519 | 32 | 32 | 64 |

We are assuming an online $\tau$-times signature scheme. Orig. compression uses the same format at Falcon, alt. compression uses different parameters that optimise for our larger parameters.

**Table 3.** Benchmark results of comparable signature schemes on an ARM Cortex A53.

| | Clock Cycles | | | Runtime (ms) | | |
|---|---|---|---|---|---|---|
| Signature Scheme | KeyGen | Sign | Verify | KeyGen | Sign | Verify |
| Falcon-512-EMU [PFH+22] | 153M | 35M | 172K | 255.09 | 58.65 | 0.28 |
| This work (EMU, offline) | 153M | 6M | - | 255.09 | 9.45 | - |
| This work (EMU, online) | - | 10M | 172K | - | 16.16 | 0.28 |
| Falcon-512-FPU [PFH+22] | 87M | 3M | 171K | 144.93 | 4.73 | 0.28 |
| This work (FPU, offline) | 87M | 1M | - | 144.93 | 1.59 | - |
| This work (FPU, online) | - | 0.8M | 171K | - | 1.29 | 0.28 |
| Dilithium2 [LDK+22] | 1.6M | 12.5M | 1.7M | 2.67 | 20.87 | 2.82 |
| Ed25519 [Pet17] | 0.5M | 0.5M | 1.4M | 0.81 | 0.84 | 2.36 |

GCC version 10.2.1-6+rpi1 for compilation, targeting the ARMv7l (AArch32) architecture. We run the benchmarks on the slowest possible clock speed and only use a single core. The benchmarks are run over 25,000 repetitions and incorporate sleep states where necessary. Overall, these settings all ensure the most fair and accurate results on the embedded device.

For Falcon and our construction we also present benchmarks for both single-precision and double-precision floating-point operations, the former emulates the 53 bits of floating-point precision required, and the latter does this natively by utilising the double precision FPU. One of the particular reasons we chose the ARM Cortex A53 was to observe the contrast between native and emulated floating-point operations. Since previous research on similar low-cost devices like the ARM Cortex M7 have shown non-constant runtimes for Falcon [HW23], thus we decided to run on the ARM Cortex A53 instead.

In order to fairly compare benchmarks, we also include performance results for Dilithium and Falcon (both using the latest version of their reference implementations) as well as Ed25519 (taken from [Pet17]).

We present our benchmarks using cycle counts as well as runtime (in milliseconds). In Table 3 we present the benchmarks for each of the signature primitives

**Table 4.** Categorising ARM Cortex A53 benchmarks for offline/online computations.

| Signature Scheme | Clock Cycles | | Runtime (ms) | |
|---|---|---|---|---|
| | Offline | Online | Offline | Online |
| Falcon-512-EMU [PFH+22] | 153 048 368 | 35 362 439 | 255.09 | 58.94 |
| This work, EMU | 158 718 728 | 10 042 709 | 264.53 | 16.73 |
| Falcon-512-FPU [PFH+22] | 86 957 688 | 2 991 232 | 144.93 | 5.01 |
| This work, FPU | 87 916 636 | 1 118 903 | 146.53 | 1.86 |
| Dilithium2 [LDK+22] | 1 604 583 | 14 216 595 | 2.67 | 23.69 |
| Ed25519 [Pet17] | 483 962 | 1 920 361 | 0.81 | 3.20 |

and in Table 4, we present these results using the offline and online separations, all of these are presented for our construction alongside Dilithium2, Falcon-512, and Ed25519, for comparison.

The overall design of this work aims to improve the online cost in comparison to current PQC signature schemes and potentially classical schemes in use today. With respect to offline costs, our construction uses the same key generation procedure and offloads 5.6m cycles from its signature generation to the offline part, thus increasing the clock cycles required offline by less than 1–4%[3]. These extra offline costs mainly contain FFT transformations of four polynomials, discrete Gaussian sampling of two polynomials, and an NTT calculation of a target polynomial. Thus, there is only a small performance difference between Falcon and our offline costs. When compared to Dilithium and Ed25519 it is orders of magnitude slower, but this is likely going to be the case on every platform and thus is due to the algorithms themselves.

Our online costs still consist of FFT conversions, in particular those needed when computing the short preimage, but the module overall is significantly less involved than the preimage computation in Falcon. The remaining operations required in Sign-Online are some randomness generation, hashing of the message, and the compression of the signature. These remaining operations utilise the same functions used in Falcon.

Considering that this proof-of-concept implementation is not highly optimised, the results are promising in comparison to the NIST PQC standards and to current classical standards like Ed25519.

When comparing the online versions of these signature schemes, the reader is reminded that for our scheme (and any scheme that utilises the online/offline paradigm [EGM90,EGM96]) it requires one few-time signature generation and two verifications (one of the long-term signature and one of the few-time signature); the regular signature schemes' costs are simply the cost of one signature generation and one signature verification.

In comparison to the signature generation costs, our construction requires 3.6x less clock cycles compared to Falcon, which reduces to 3.5x when the full online costs are considered with the additional signature verification. We also see that our construction requires fewer clock cycles than Dilithium2, using both

[3] Depending on whether the floating-point arithmetic is emulated or uses the FPU.

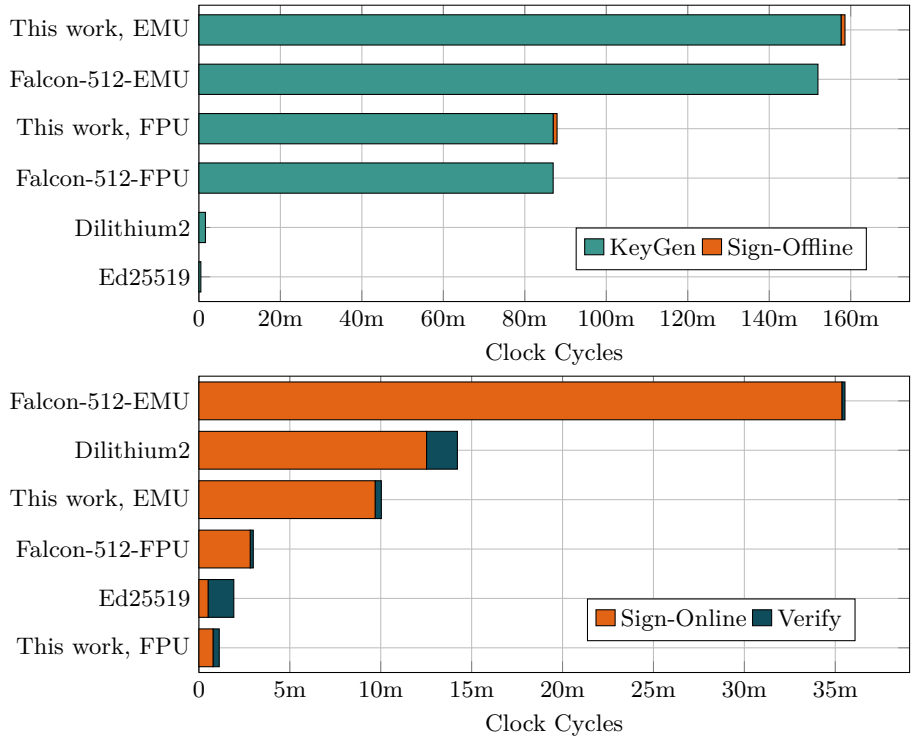


**Fig. 8.** Key generation, signature generation (offline and online), and verify benchmarks on ARM Cortex A53. Top graph consists of total offline cost, key generation + sign offline cycles. Bottom graph consists of total online cost, signature generation and verification cycles.

native and emulated floating-point arithmetic, and even competes with Ed25519 when using native floats.

When compared to Ed25519 for overall online costs, Falcon's fast verification means we gain advantages when considering total signature and verification costs with Ed25519 requiring just under 2m clock cycles and our construction requiring just over 1m cycles, overall reducing cycles needed by 58%.

## References

AAC+22. Gorjan Alagic, Daniel Apon, David Cooper, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Carl Miller, Dustin Moody, Rene Peralta, et al. NIST IR 8413: Status report on the third round of the NIST post-quantum cryptography standardization process. *US Department of Commerce, NIST*, 2022. 1

ABD16. Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded

encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 153–178. Springer, Berlin, Heidelberg, August 2016. 2.5

APS15. Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of Learning with Errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015. 4, A

Bab86. László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986. 2, 2

BBE+19. Gilles Barthe, Sonia Belaïd, Thomas Espitau, Pierre-Alain Fouque, Mélissa Rossi, and Mehdi Tibouchi. GALACTICS: Gaussian sampling for lattice-based constant- time implementation of cryptographic signatures, revisited. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2147–2164. ACM Press, November 2019. 4.1

BLR+18. Shi Bai, Tancrède Lepoint, Adeline Roux-Langlois, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance. *Journal of Cryptology*, 31(2):610–640, April 2018. 1, 1.1, 2

DLP14. Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over NTRU lattices. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 22–41. Springer, Berlin, Heidelberg, December 2014. 1

DN12. Léo Ducas and Phong Q. Nguyen. Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 433–450. Springer, Berlin, Heidelberg, December 2012. 1.1, 1

DP16. Léo Ducas and Thomas Prest. Fast Fourier Orthogonalization. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, pages 191–198, 2016. 1, 2, 2.5

DPPvW22. Léo Ducas, Eamonn W. Postlethwaite, Ludo N. Pulles, and Wessel P. J. van Woerden. Hawk: Module LIP makes lattice signatures fast, compact and simple. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 65–94. Springer, Cham, December 2022. 1

DvW21. Léo Ducas and Wessel P. J. van Woerden. NTRU fatigue: How stretched is overstretched? In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 3–32. Springer, Cham, December 2021. 2.5

EFG+22. Thomas Espitau, Pierre-Alain Fouque, François Gérard, Mélissa Rossi, Akira Takahashi, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu. Mitaka: A simpler, parallelizable, maskable variant of falcon. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 222–253. Springer, Cham, May / June 2022. 1

EGM90. Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital schemes. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 263–275. Springer, New York, August 1990. 1, 2.3, 3, 4.1

EGM96. Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, March 1996. 1, 2.3, 3, 4.1

GGH97. Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski, Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 112–131. Springer, Berlin, Heidelberg, August 1997. 1

GLP12. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547. Springer, Berlin, Heidelberg, September 2012. 4.1

GM18. Nicholas Genise and Daniele Micciancio. Faster Gaussian sampling for trapdoor lattices with arbitrary modulus. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 174–203. Springer, Cham, April / May 2018. 2.4

GMRR22. Morgane Guerreau, Ange Martinelli, Thomas Ricosset, and Mélissa Rossi. The hidden parallelepiped is back again: Power analysis attacks on falcon. *IACR TCHES*, 2022(3):141–164, 2022. 1

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 1, 2.4, A

HHP+03. Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In Marc Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 122–140. Springer, Berlin, Heidelberg, April 2003. 1

HW23. James Howe and Bas Westerbaan. Benchmarking and analysing the NIST PQC lattice-based signature schemes standards on the ARM Cortex M7. In Nadia El Mrabet, Luca De Feo, and Sylvain Duquesne, editors, *AFRICACRYPT 23*, volume 14064 of *LNCS*, pages 442–462. Springer, Cham, July 2023. 1, 4.1

KA21. Emre Karabulut and Aydin Aysu. FALCON down: Breaking FALCON post-quantum signature scheme through side-channel attacks. In *58th ACM/IEEE Design Automation Conference, DAC 2021, San Francisco, CA, USA, December 5-9, 2021*, pages 691–696. IEEE, 2021. 1

KF17. Paul Kirchner and Pierre-Alain Fouque. Revisiting lattice attacks on overstretched NTRU parameters. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 3–26. Springer, Cham, April / May 2017. 2.5

LDK+22. Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. 3, 4

LS15. Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *DCC*, 75(3):565–599, 2015. 2.5

LTYZ24. Xiuhan Lin, Mehdi Tibouchi, Yang Yu, and Shiduo Zhang. Do not disturb a sleeping falcon: Floating-point error sensitivity of the falcon sampler and its consequences. Cryptology ePrint Archive, Paper 2024/1709, 2024. 1

MGE91. Silvio Micali, Oded Goldreich, and Shimon Even. On-line/off-line digital signing. U.S. Patent #5,016,274, May 1991. https://patents.google.com/patent/US5016274A/en. 2.3

ML-23a. FIPS 204 (Initial Public Draft): Module-Lattice-Based Digital Signature Standard. National Institute of Standards and Technology, NIST FIPS PUB 204, U.S. Department of Commerce, August 2023. 1

ML-23b. FIPS 203 (Initial Public Draft): Module-Lattice-Based Key-Encapsulation Mechanism Standard. National Institute of Standards and Technology, NIST FIPS PUB 203, U.S. Department of Commerce, August 2023. 1

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Berlin, Heidelberg, April 2012. 2.4

MR04. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381. IEEE Computer Society Press, October 2004. 2, 3.1, 3.1

NR06. Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 271–288. Springer, Berlin, Heidelberg, May / June 2006. 1, 1.1, 1

Pei10. Chris Peikert. An efficient and parallel Gaussian sampler for lattices. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 80–97. Springer, Berlin, Heidelberg, August 2010. 4.1

Pet17. Orson Peters. Ed25519. https://github.com/orlp/ed25519, 2017. commit: b1f19fab4aebe607805620d25a5e42566ce46a0e. 3, 4.1, 4

PFH+22. Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. 1, 2.5, 4, 3, 4

Por19. Thomas Pornin. New efficient, constant-time implementations of Falcon. Cryptology ePrint Archive, Report 2019/893, 2019. 4.1

SLH23. FIPS 205 (Initial Public Draft): Stateless Hash-Based Digital Signature Standard. National Institute of Standards and Technology, NIST FIPS PUB 205, U.S. Department of Commerce, August 2023. 1

SS11. Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Berlin, Heidelberg, May 2011. 1, 2.5

ST01. Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 355–367. Springer, Berlin, Heidelberg, August 2001. 2.3, 3

ZLYW23. Shiduo Zhang, Xiuhan Lin, Yang Yu, and Weijia Wang. Improved power analysis attacks on falcon. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 565–595. Springer, Cham, April 2023. 1

## A Proof of Theorem 1

*Proof.* We start the game hop proof with the EUF-CMA game specialised to the GPV-Online/Offline signatures in the ROM as Game 0, illustrated in Figure 9.

To ease notation, we leave the bookkeeping of the number of queries implicit in the games that follow, i.e. we restrict to adversaries that only submit up to $Q$ queries.

$\mathsf{Game}_1$*: Programming the random oracle.* In the first hop, we program the random oracle to answer hash queries as follows. Given a $(\mu, \mathbf{r})$ hash query, look up if $(\mu, \mathbf{r})$ was previously queried. If so, answer as before. Otherwise, sample $\mathbf{y} \sim \mathcal{D}_{\mathbb{Z}^m, s_{\mathsf{sim}}}$, using a discrete Gaussian sampler $\mathsf{SampD}(1^n, 1^m, s_{\mathsf{sim}})$ where $s_{\mathsf{sim}}^2 = r \cdot s^2$, for some positive real parameter $r \geq 1$, store $\mathbf{y}^{(s)}$ in a pre-image list as $\mathcal{P}(\mu, \mathbf{r}) \leftarrow \mathbf{y}$, and programme the random oracle $\mathsf{RO}$ at $(\mu, \mathbf{r})$ as $\mathsf{RO}(\mu, \mathbf{r}) \leftarrow \mathbf{A} \cdot \mathbf{y} \bmod q$. For $s_{\mathsf{sim}}$ exceeding the smoothing parameter $\eta_\epsilon(\Lambda^*(\mathbf{A}))$, by [GPV08, Lem 5.2], the syndrome $\mathbf{A} \cdot \mathbf{y} \bmod q$ is within statistical distance $2\epsilon$ of the uniform distribution in $\mathbb{Z}_q^n$. Therefore, $\mathsf{Game}_0$ and $\mathsf{Game}_1$ distributions are statistically close and hence indistinguishable. Therefore the adversary's success probability may only increase by a negligible amount, accounted for in the $Q \cdot \mathsf{negl}(\lambda)$ term in the statement of the theorem.

$\mathsf{Game}_2$*: checking message-salt pair repeated queries.* The next game hop incorporates a check to ensure that during a signature query, the salt drawn by the signing algorithm is not a repetition of a salt that was queried before along with the same message. We justify this hop from $\mathsf{Game}_1$ to $\mathsf{Game}_2$ by invoking the fundamental lemma of game playing, which assures that $\Pr[repeat] \leq |Q|/2^{\lambda/2}$, so the distance in this hop is again accounted for in the the $Q \cdot \mathsf{negl}(\lambda)$ term.

$\mathsf{Game}_3$*: simulating signatures.* We answer the signature query as depicted in Figure 10. In particular, the random oracle is called on the input message and the randomly drawn salt, and the returned pre-image $y$ is appended with the salt and set as the signature. We prove that the probability that a $Q$-query adversary wins $\mathsf{Game}_2$ is at most the probability of winning $\mathsf{Game}_3$, up to the multiplicative factor on the right side of Lemma 1.

$\mathsf{Game}_4$*: forget the trapdoor.* The final game hop forgets the trapdoor and in place of the matrix $A$ associated with the trapdoor, draws a random $\mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m}$. Since the trapdoor generation algorithm generates $\mathbf{A}$ which are pseudorandom, the distribution of $\mathsf{Game}_3$ and $\mathsf{Game}_4$ are indistinguishable.

*Constructing a SIS adversary.* An adversary that wins $\mathsf{Game}_4$ can be translated into one that solves the SIS problem as follows. To win $\mathsf{Game}_4$, the adversary has to output a $(\mu^\star, (\mathbf{y}^\star, \mathbf{r}^\star))$ such that $H(\mu^\star, \mathbf{r}^\star) = \mathbf{A} \cdot \mathbf{y}$ and $\|\mathbf{y}^\star\|_2 \leq \beta$. Without querying for $H(\mu^\star, \mathbf{r}^\star)$, the adversary does not know the nearly uniformly random target, and therefore can succeed in guessing with a negligible probability $1/q^n$. Therefore, we may assume that adversary indeed queried for $t := H(\mu^\star, \mathbf{r}^\star)$. The randomness of $y$ from the pre-image sampling algorithm ensures that with high probability $\mathbf{y} \neq \mathbf{y}^\star$. Therefore, the non-zero short vector $\mathbf{y} - \mathbf{y}^\star$ is in the dual of the lattice generated by $\mathbf{A}$, since $\mathbf{A} \cdot \mathbf{y} = \mathbf{A} \cdot \mathbf{y}^\star = H(\mu^\star, \mathbf{r}^\star) \bmod q \Rightarrow \mathbf{A} \cdot (\mathbf{y} - \mathbf{y}^\star) = 0 \mod q$. Further, it is a short lattice vector, since $\|\mathbf{y} - \mathbf{y}^\star\|_2 \leq 2\beta$.

$\mathsf{Game}_0$ / $\mathsf{SIGN}(\mu)$ / $\mathsf{RO}(\mu, \mathbf{r})$

$\mathcal{Q}, \mathcal{H}, c \leftarrow \emptyset, \emptyset, 0$
$(\mathbf{A}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$
$\{(\mathbf{x}^i_{\mathrm{pre}}, \mathbf{t}^i_{\mathrm{pre}}) \leftarrow \Sigma.\mathsf{PreSign}(\mathsf{td})\}_{1 \le i \le Q}$
$(\mu^\star, (\mathbf{y}^\star, \mathbf{r}^\star)) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_Q(\mathbf{A})$
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$
$b_1 := \|\mathbf{y}^\star\| \le \beta$
$b_2 := \mathcal{H}[\mu, \mathbf{r}] = \mathbf{A} \cdot \mathbf{y}^\star$
**return** $b_0 \wedge b_1 \wedge b_2$

**if** $c \ge Q$ **then abort**
$\mathbf{r} \leftarrow\!\$\ \{0,1\}^\lambda$
$\mathbf{t} \leftarrow \mathsf{RO}(\mu, \mathbf{r})$
$\mathbf{x} \leftarrow \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathbf{t} - \mathbf{t}^c_{\mathrm{pre}})$
$\mathbf{y} \leftarrow \mathbf{x} + \mathbf{x}^c_{\mathrm{pre}}$
$\sigma \leftarrow (\mathbf{y}, \mathbf{r})$
$c \leftarrow c + 1$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$
**return** $\sigma$

**if** $(\mu, \mathbf{r}) \notin \mathcal{H}$ **then**
  $\mathcal{H}[\mu, \mathbf{r}] \leftarrow\!\$\ \mathbb{Z}_q^n$
**return** $\mathcal{H}[\mu, \mathbf{r}]$

$\mathsf{Game}_1$ / $\mathsf{SIGN}(\mu)$ / $\mathsf{RO}(\mu, \mathbf{r})$

$\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$
$(\mathbf{A}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$
$\{(\mathbf{x}^i_{\mathrm{pre}}, \mathbf{t}^i_{\mathrm{pre}}) \leftarrow \Sigma.\mathsf{PreSign}(\mathsf{td})\}_{1 \le i \le Q}$
$(\mu^\star, (\mathbf{y}^\star, \mathbf{r}^\star)) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_Q(\mathbf{A})$
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$
$b_1 := \|\mathbf{y}^\star\| \le \beta$
$b_2 := \mathcal{H}[\mu, \mathbf{r}] = \mathbf{A} \cdot \mathbf{y}^\star$
**return** $b_0 \wedge b_1 \wedge b_2$

**if** $c \ge Q$ **then abort**
$\mathbf{r} \leftarrow\!\$\ \{0,1\}^\lambda$
$\mathbf{t} \leftarrow \mathsf{RO}(\mu, \mathbf{r})$
$\mathbf{x} \leftarrow \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathbf{t} - \mathbf{t}^c_{\mathrm{pre}})$
$\mathbf{y} \leftarrow \mathbf{x} + \mathbf{x}^c_{\mathrm{pre}}$
$\sigma \leftarrow (\mathbf{y}, \mathbf{r})$
$c \leftarrow c + 1$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$
**return** $\sigma$

**if** $(\mu, \mathbf{r}) \notin \mathcal{H}$ **then**
  $\mathbf{y} \leftarrow\!\$\ \mathsf{SampD}(1^n, 1^m, s_{\mathsf{sim}})$
  $\mathbf{t} := \mathbf{A} \cdot \mathbf{y} \bmod q$
  $\mathcal{H}(\mu, \mathbf{r}) \leftarrow \mathbf{t}$
  $\mathcal{P}(\mu, \mathbf{r}) \leftarrow \mathbf{y}$
**return** $\mathcal{H}[\mu, \mathbf{r}]$

$\mathsf{Game}_2$ / $\mathsf{SIGN}(\mu)$ / $\mathsf{RO}(\mu, \mathbf{r})$

$\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$
$\mathit{repeat} \leftarrow \mathsf{false}$
$(\mathbf{A}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$
$\{(\mathbf{x}^i_{\mathrm{pre}}, \mathbf{t}^i_{\mathrm{pre}}) \leftarrow \Sigma.\mathsf{PreSign}(\mathsf{td})\}_{1 \le i \le Q}$
$(\mu^\star, (\mathbf{y}^\star, \mathbf{r}^\star)) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_Q(\mathbf{A})$
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$
$b_1 := \|\mathbf{y}^\star\| \le \beta$
$b_2 := \mathcal{H}[\mu, \mathbf{r}] = \mathbf{A} \cdot \mathbf{y}^\star$
**return** $b_0 \wedge b_1 \wedge b_2$

**if** $c \ge Q$ **then abort**
$\mathbf{r} \leftarrow\!\$\ \{0,1\}^\lambda$
**if** $(\mu, \mathbf{r}) \in \mathcal{H}$
  $\mathit{repeat} \leftarrow \mathsf{true}$
  **abort**
$\mathbf{t} \leftarrow \mathsf{RO}(\mu, \mathbf{r})$
$\mathbf{x} \leftarrow \mathsf{SampPre}_{\gg}(\mathsf{td}, \mathbf{t} - \mathbf{t}^c_{\mathrm{pre}})$
$\mathbf{y} \leftarrow \mathbf{x} + \mathbf{x}^c_{\mathrm{pre}}$
$\sigma \leftarrow (\mathbf{y}, \mathbf{r})$
$c \leftarrow c + 1$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$
**return** $\sigma$

**if** $(\mu, \mathbf{r}) \notin \mathcal{H}$ **then**
  $\mathbf{y} \leftarrow\!\$\ \mathsf{SampD}(1^n, 1^m, s_{\mathsf{sim}})$
  $\mathbf{t} := \mathbf{A} \cdot \mathbf{y} \bmod q$
  $\mathcal{H}(\mu, \mathbf{r}) \leftarrow \mathbf{t}$
  $\mathcal{P}(\mu, \mathbf{r}) \leftarrow \mathbf{y}$
**return** $\mathcal{H}[\mu, \mathbf{r}]$

**Fig. 9.** $\mathsf{Game}_0$ to $\mathsf{Game}_2$

The only hop that contributes non negligibly to the bit security claimed in the theorem is the jump from $\mathsf{Game}_2$ to $\mathsf{Game}_3$, which Lemma 1 bounds as

$$\frac{\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_2}(\mathsf{vk})=1\right]}{\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_3}(\mathsf{vk})=1\right]^{\frac{a-1}{a}}} \leq 2^{\left(\log_2\left(\frac{s_{\mathsf{sim}}^2}{s^2}\right)+\frac{(a-1)\pi\|\mathbf{x}\|_{\max}^2\log_2 e}{\left(a\left(\frac{s_{\mathsf{sim}}^2}{s^2}-1\right)+1\right)s^2}\right)Q}. \tag{9}$$

Here, we ignored the $(1+\epsilon)^Q$ by choosing a small enough parameter $\epsilon = 2^{-128}$. We have the two parameters $a$ and $r$ to optimize this bound. We will pick $a$ to be big enough to ensure $(a-1)/a$ is close to one. By choosing the largest $a$ satisfying the constraint $s_{\mathsf{sim}} \geq \sqrt{\ln(2m(1+1/\epsilon)/\pi)}\cdot\sqrt{(a((s_{\mathsf{sim}}^2/s^2)-1)+1)}$ of Lemma 1, we may approximate $(a-1)/a \approx 1$ and $(a-1)/(a(s_{\mathsf{sim}}^2/s^2-1))+1) \approx 1/(s_{\mathsf{sim}}^2/s^2-1)$. Hence, Equation (9) simplifies to

$$\begin{aligned}-\log_2\left(\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_2}(\mathsf{vk})=1\right]\right) \leq \\ -\log_2\left(\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_3}(\mathsf{vk})=1\right]\right) - \left(\log_2\left(\frac{s_{\mathsf{sim}}^2}{s^2}\right)+\frac{\pi\|\mathbf{x}\|_{\max}^2\log_2 e}{\left(\frac{s_{\mathsf{sim}}^2}{s^2}-1\right)s^2}\right)Q.\end{aligned} \tag{10}$$

We bound $\log_2\left(\Pr\left[\mathcal{A}_Q^{\mathsf{Game}_4}(\mathsf{vk})=1\right]\right)$ empirically using the lattice estimator [APS15], which considers the best known cryptanalytic algorithms for forgery by solving SIS. This estimate depends on the choice of the simulator Gaussian parameter $s_{\mathsf{sim}}$. We can choose the $s_{\mathsf{sim}}^2/s^2$ to balance the two terms on Equation (10). One simple choice (which may not be asymptotically optimal) is to match the simulator and signature variances by setting $\frac{s_{\mathsf{sim}}^2}{2\pi} = \frac{s^2}{2\pi} + \mathbb{E}_{\mathbf{x}}\left(\|\mathbf{x}\|^2\right)$, resulting in the right-most term in Equation (10) being

$$\left(\log_2\left(1+\frac{2\pi\mathbb{E}_{\mathbf{x}}\left(\|\mathbf{x}\|^2\right)}{s^2}\right)+\frac{\|\mathbf{x}\|_{\max}^2\log_2 e}{2\mathbb{E}_{\mathbf{x}}\left(\|\mathbf{x}\|^2\right)}\right)\cdot Q. \tag{11}$$

Another choice is the higher value $\frac{s_{\mathsf{sim}}^2}{2\pi} := \frac{s^2}{2\pi} + \|\mathbf{x}\|_{\max}^2$, resulting in the right-most term in Equation (10) being

$$\left(\log_2\left(1+\frac{2\pi\|\mathbf{x}\|_{\max}^2}{s^2}\right)+\frac{\log_2 e}{2}\right)\cdot Q, \tag{12}$$

which is smaller than Equation (11). Substituting, Equation (11) and Equation (12) into Equation (10), proves the bound. □

| $\mathsf{Game}_3$ | $\mathsf{SIGN}(\mu)$ | $\mathsf{RO}(\mu, \mathbf{r})$ |
|---|---|---|
| $\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$ | **if** $c \geq Q$ **then abort** | **if** $(\mu, \mathbf{r}) \notin \mathcal{H}$ **then** |
| $repeat \leftarrow \mathsf{false}$ | $\mathbf{r} \leftarrow\!\!\$\ \{0,1\}^\lambda$ | $\mathbf{y} \leftarrow\!\!\$\ \mathsf{SampD}(1^n, 1^m, s_{\mathsf{sim}})$ |
| $(\mathbf{A}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ | **if** $(\mu, \mathbf{r}) \in \mathcal{H}$ | $\mathbf{t} := \mathbf{A} \cdot \mathbf{y} \bmod q$ |
| $\{(\mathbf{x}^i_{\mathrm{pre}}, \mathbf{t}^i_{\mathrm{pre}}) \leftarrow \Sigma.\mathsf{PreSign}(\mathsf{td})\}_{1 \leq i \leq Q}$ | $repeat \leftarrow \mathsf{true}$ | $\mathcal{H}(\mu, \mathbf{r}) \leftarrow \mathbf{t}$ |
| $(\mu^\star, (\mathbf{y}^\star, \mathbf{r}^\star)) \leftarrow \mathcal{A}_Q^{\mathsf{SIGN},\mathsf{RO}}(\mathbf{A})$ | **abort** | $\mathcal{P}(\mu, \mathbf{r}) \leftarrow \mathbf{y}$ |
| $b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$ | $\mathsf{RO}(\mu, \mathbf{r})$ | **return** $\mathcal{H}[\mu, \mathbf{r}]$ |
| $b_1 := \|\mathbf{y}^\star\| \leq \beta$ | $\mathbf{y} \leftarrow \mathcal{P}[\mu, \mathbf{r}]$ | |
| $b_2 := \mathcal{H}[\mu, \mathbf{r}] = \mathbf{A} \cdot \mathbf{y}^\star$ | $\sigma \leftarrow (\mathbf{y}, \mathbf{r})$ | |
| **return** $b_0 \wedge b_1 \wedge b_2$ | $c \leftarrow c + 1$ | |
| | $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$ | |
| | **return** $\sigma$ | |

| $\mathsf{Game}_4$ | $\mathsf{SIGN}(\mu)$ | $\mathsf{RO}(\mu, \mathbf{r})$ |
|---|---|---|
| $\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$ | **if** $c \geq Q$ **then abort** | **if** $(\mu, \mathbf{r}) \notin \mathcal{H}$ **then** |
| $repeated \leftarrow \mathsf{false}$ | $\mathbf{r} \leftarrow\!\!\$\ \{0,1\}^\lambda$ | $\mathbf{y} \leftarrow\!\!\$\ \mathsf{SampD}(1^n, 1^m, s_{\mathsf{sim}})$ |
| $\mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m}$ | **if** $(\mu, \mathbf{r}) \in \mathcal{H}$ | $\mathbf{t} := \mathbf{A} \cdot \mathbf{y} \bmod q$ |
| $(\mu^\star, (\mathbf{y}^\star, \mathbf{r}^\star)) \leftarrow \mathcal{A}_Q^{\mathsf{SIGN},\mathsf{RO}}(\mathbf{A})$ | $repeat \leftarrow \mathsf{true}$ | $\mathcal{H}(\mu, \mathbf{r}) \leftarrow \mathbf{t}$ |
| $\{(\mathbf{x}^i_{\mathrm{pre}}, \mathbf{t}^i_{\mathrm{pre}}) \leftarrow \Sigma.\mathsf{PreSign}(\mathsf{td})\}_{1 \leq i \leq Q}$ | **abort** | $\mathcal{P}(\mu, \mathbf{r}) \leftarrow \mathbf{y}$ |
| $b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$ | $\mathsf{RO}(\mu, \mathbf{r})$ | **return** $\mathcal{H}[\mu, \mathbf{r}]$ |
| $b_1 := \|\mathbf{y}^\star\| \leq \beta$ | $\mathbf{y} \leftarrow \mathcal{P}[\mu, \mathbf{r}]$ | |
| $b_2 := \mathcal{H}[\mu, \mathbf{r}] = \mathbf{A} \cdot \mathbf{y}^\star$ | $\sigma \leftarrow (\mathbf{y}, \mathbf{r})$ | |
| **return** $b_0 \wedge b_1 \wedge b_2$ | $c \leftarrow c + 1$ | |
| | $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$ | |
| | **return** $\sigma$ | |

**Fig. 10.** $\mathsf{Game}_3$ and $\mathsf{Game}_4$