Practical Signatures from the Partial Fourier Recovery Problem

Jeff Hoffstein[1], Jill Pipher[1], John M. Schanck[2], Joseph H. Silverman[1], and William Whyte[2]

[1] Brown University, Providence, RI, 02912
{jhoff,jpipher,jhs}@math.brown.edu
[2] Security Innovation, Wilmington, MA 01887
{jschanck,wwhyte}@securityinnovation.com

Abstract. We present PASS_{RS} , a variant of the prior PASS and PASS-2 proposals, as a candidate for a practical post-quantum signature scheme. Its hardness is based on the problem of recovering a ring element with small norm from an incomplete description of its Chinese remainder representation. For our particular instantiation, this corresponds to the recovery of a vector with small infinity norm from a limited set of its Fourier coefficients.

The key improvement over previous versions of PASS is the introduction of a rejection sampling technique from Lyubashevsky (2009) which assures that transcript distributions are completely decoupled from the keys that generate them.

Although the scheme is not supported by a formal security reduction, we present extensive arguments for its security and derive concrete parameters based on the performance of state of the art lattice reduction and enumeration techniques.

1 Introduction

In the late 1990s two authors of the present paper proposed authentication and signature schemes based on the problem of recovering a polynomial with tightly concentrated coefficients given a small number of evaluations of that polynomial. The heuristic justification for the security of the scheme was that the uncertainty principle severely restricts how concentrated a signal can be in two mutually incoherent bases.

An early incarnation of the scheme is found in [12], and a later version, called PASS-2 was published in [13]. A rough description goes as follows. Let N be a positive integer, and choose a prime $q = rN+1$, with $r \geq 1$. We will denote by R_q the ring $\mathbb{Z}_q[x]/(x^N - 1)$, though we will often treat elements of R_q as vectors in $\mathbb{Z}_q^N$ equipped with the $\star$-multiplication of R_q. To avoid confusion, we will denote component-wise multiplication of vectors by $\odot$. For any β, with $(\beta, q) = 1$, it follows from Fermat's little theorem that $\beta^{rN} \equiv 1 \pmod{q}$. Consequently, the mapping $\boldsymbol{f} \to \boldsymbol{f}(\beta^r)$ is well defined for any $\boldsymbol{f}$ in R_q. In addition to being well defined, it is also a ring homomorphism, for the simple reason that for any $\boldsymbol{f}_1, \boldsymbol{f}_2 \in R_q$,

$$(\boldsymbol{f}_1 + \boldsymbol{f}_2)(\beta^r) = \boldsymbol{f}_1(\beta^r) + \boldsymbol{f}_2(\beta^r) \quad \text{and} \quad (\boldsymbol{f}_1 \star \boldsymbol{f}_2)(\beta^r) = \boldsymbol{f}_1(\beta^r)\boldsymbol{f}_2(\beta^r).$$

More generally, for any $\Omega = \{\beta_1^r, \beta_2^r, \ldots, \beta_t^r\}$, the mapping $\mathcal{F} : R_q \to \mathbb{Z}_q^t$ given by

$$\mathcal{F}_\Omega \boldsymbol{f} = (\boldsymbol{f}(\beta_1^r), \boldsymbol{f}(\beta_2^r), \ldots, \boldsymbol{f}(\beta_t^r))^T$$

is a ring homomorphism, with addition and $\odot$-multiplication modulo q done on the right hand side. This is an example of the more general phenomenon of the ring homomorphism mapping functions to their Fourier transforms.

In the above setting, the uncertainty principle implies that a ring element with a coefficient vector drawn from a small region of $\mathbb{Z}_q^N$ will have widely dispersed discrete Fourier coefficients. For instance a vector with small infinity norm, e.g. with coefficients in $\{-1, 0, 1\}$, will likely be supported on all powers of a primitive N^{th} root ω and will have Fourier coefficients which are essentially uniformly distributed in $\mathbb{Z}_q$.

The hard problem in PASS can be stated as the following underdetermined linear inversion problem, which we will refer to as the *partial Fourier recovery* problem. Let ω be a primitive N^{th} root of unity modulo q. We define the discrete Fourier transform over $\mathbb{Z}_q$ to be the linear transformation $\mathcal{F}\boldsymbol{f} = \widehat{\boldsymbol{f}} : \mathbb{Z}_q^N \to \mathbb{Z}_q^N$ given by

$$(\mathcal{F})_{i,j} = \omega^{ij}.$$

Furthermore, let $\mathcal{F}_\Omega$ be the restriction of $\mathcal{F}$ to the set of t rows specified by an index set Ω,

$$(\mathcal{F}_\Omega)_{i,j} = \omega^{\Omega_i j}.$$

The partial Fourier recovery problem is: given an evaluation $\widehat{\boldsymbol{f}}|_\Omega \in \mathbb{Z}_q^t$, find $\boldsymbol{x}$ with small norm such that $\widehat{\boldsymbol{x}}|_\Omega = \widehat{\boldsymbol{f}}|_\Omega \pmod q$.

The problem of recovering a signal from a restricted number of its Fourier coefficients is well studied and known to be quite difficult in general. The restricted image $\widehat{\boldsymbol{f}}|_\Omega$ is expected to contain very little information about the unobserved Fourier coefficients (the evaluations of $\boldsymbol{f}$ on ω^i for i not in Ω), and often the only way to recover $\boldsymbol{f}$ will be an expensive combinatorial optimization procedure. However, there are cases (some quite surprising) in which the problem is known to be easy.

Certainly, if $t \log q$ is small, brute force search over $\boldsymbol{f}'$ with appropriate norm may be a viable solution – each randomly chosen candidate having essentially a q^{-t} chance of evaluating to $\widehat{\boldsymbol{f}}|_\Omega$.

The problem is trivial in the large t regime, $t \geq N$, since any rank N submatrix of the chosen Vandermonde matrix will be invertible. As t decreases slightly below N, or we allow some portion of the coefficients to be corrupted, the problem essentially becomes that of decoding Reed-Solomon codes and we can expect to recover $\boldsymbol{f}$ by list-decoding or similar techniques. Efficient recovery of general signals when t is much less than N would have significant coding theoretic implications.

For t in an intermediate range, say $t \approx N/2$, the situation is more complicated. Were one to consider the complex Fourier transform rather than the number theoretic transform, one might be able to apply techniques from the field of compressed sensing. Recent work in this field has delineated cases in which a *sparse* signal can be recovered from a limited number of its (complex) Fourier coefficients by an L^1 optimization procedure. For this to be successful the signals must be very sparse, having a number of non-zero coefficients which is less than $|\Omega|/2$ [2]. It is not clear how these results translate into the finite field setting.

As far as we are aware, the best technique for solving the partial Fourier recovery problem is by solving an associated closest vector problem. Specifically, let $\varLambda^{\perp}(\mathcal{F}_\Omega)$ be the lattice of vectors in the kernel of $\mathcal{F}_\Omega$. That is,

$$\varLambda^{\perp}(\mathcal{F}_\Omega) = \left\{ \boldsymbol{a} \in \mathbb{Z}_q^N \ : \ \mathcal{F}_\Omega \boldsymbol{a} = \boldsymbol{0} \pmod{q} \right\}.$$

If, given $\boldsymbol{y} \in \mathbb{Z}_q^N$, a point $\boldsymbol{x} \in \varLambda^{\perp}(\mathcal{F}_\Omega)$ can be found such that $\|\boldsymbol{y} - \boldsymbol{x}\|_\infty \leq \beta$, then $\mathcal{F}_\Omega(\boldsymbol{y} - \boldsymbol{x}) = \widehat{\boldsymbol{y}}|_\Omega$ and $\|\boldsymbol{y} - \boldsymbol{x}\|_\infty \leq \beta$. Since one can easily find (large) $\boldsymbol{y}$ such that $\widehat{\boldsymbol{y}}|_\Omega = \widehat{\boldsymbol{f}}|_\Omega$ for any evaluation set $\widehat{\boldsymbol{f}}|_\Omega$, the ability to solve CVP in $\varLambda^{\perp}(\mathcal{F}_\Omega)$ implies the ability to solve arbitrary partial Fourier recovery instances

While there is no known reduction from standard lattice problems to the partial Fourier recovery problem, there is at very least a superficial relationship between finding short preimages of $\mathcal{F}_\Omega$ and another well studied hard problem. A great deal of the research in lattice based cryptography throughout the last decade has focused on a type of underdetermined linear inverse problem referred to as the small integer solution (SIS) problem.

SIS is the problem of of finding a vector $\boldsymbol{y}$ in the kernel of a specified linear transformation $\boldsymbol{A} : \mathbb{Z}_q^n \to \mathbb{Z}_q^m$ such that $\boldsymbol{y}$ is small with respect to a given norm. That is, the goal is to solve

$$\boldsymbol{A}\boldsymbol{y} = \boldsymbol{0} \pmod{q} \quad \text{and} \quad \|\boldsymbol{y}\| \leq \beta.$$

Ajtai showed in [1] that, for certain parameters and uniform random $\boldsymbol{A}$, SIS enjoys a remarkable average-case correspondence with worst-case lattice problems. That is to say that the ability to solve random SIS instances with non-negligible probability implies an ability to find short vectors in any lattice. This correspondence between worst and average cases is attractive from a provable security point of view, offering strong assurance that easy to generate instances of the SIS problem will be hard to solve, but it does not yield particularly efficient cryptosystems without additional assumptions.

The most efficient and compact SIS schemes in the literature are based on the Ideal-SIS problem, wherein the matrix $\boldsymbol{A}$ is replaced by several uniform random elements, $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots \boldsymbol{a}_k$ of a quotient ring $R_q^{\varphi} = \mathbb{Z}_q[x]/(\varphi)$. The polynomial φ is typically, but not necessarily, cyclotomic. A solution to Ideal-SIS is $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots \boldsymbol{y}_k$

in the ring such that:

$$\sum_{i=1}^{k} \boldsymbol{a}_i \star \boldsymbol{y}_i = \boldsymbol{0} \quad \text{and} \quad \sum_{i=1}^{k} \|\boldsymbol{y}_i\|^2 \leq \beta^2.$$

These schemes derive their security from the presumed hardness of Ideal-SVP – the shortest vector problem in the restricted class of lattices generated by matrix representations of elements of R_q^{φ}. Reductions from worst-case Ideal-SVP to average-case Ideal-SIS were presented in [17] [20]. Unfortunately, even with the reduced storage requirements and fast multiplication algorithms available in some rings, provably secure Ideal-SIS based constructions are still too inefficient to be competitive with existing (non-quantum resistant) schemes.

The security of PASS can be said to rest on the assumed average-case hardness of Vandermonde-SIS. We are not aware of any technique for reducing a worst-case lattice problem to Vandermonde-SIS, nor will we postulate the existence of such a reduction. We do however raise the question of whether there might be a characterization of hard instances of SIS which does not rely on structural properties of the matrix $\boldsymbol{A}$. Or more generally, when is a constrained linear inverse problem hard?

We believe an answer to this problem would likely simultaneously explain the hardness of Uniform-, Ideal- and Vandermonde-SIS, as well as delineate new classes.

2 Related Work

2.1 The original PASS protocols

Given a (padded) message μ, a secret key $\boldsymbol{f}$ with small norm, and a public key $\widehat{\boldsymbol{f}}|_{\Omega} = \mathcal{F}_{\Omega}\boldsymbol{f}$, the objective is to construct a signature that mixes $\boldsymbol{f}$ and μ and can be verified by means of $\widehat{\boldsymbol{f}}|_{\Omega}$. A prototype of this was presented in [12].

To sign, Alice

- Computes and keeps secret a short polynomial $\boldsymbol{g} \in R_q$ and reveals the commitment $\widehat{\boldsymbol{g}}|_{\Omega} = \mathcal{F}_{\Omega}\boldsymbol{g}$.
- Computes and reveals a short challenge polynomial $\boldsymbol{c} \in R_q$ from $\mathsf{Hash}(\widehat{\boldsymbol{g}}|_{\Omega}, \mu)$.
- Computes and reveals $\boldsymbol{h} = \boldsymbol{g} \star (\boldsymbol{f} + \boldsymbol{c})$.

To verify, Bob

- Verifies that $\boldsymbol{h}$ has norm less than a specific upper bound.
- Verifies that $\boldsymbol{c} = \mathsf{Hash}(\widehat{\boldsymbol{h}}|_{\Omega} / (\widehat{\boldsymbol{f}}|_{\Omega} + \widehat{\boldsymbol{c}}|_{\Omega}), \mu)$

The first condition for verification is met because

$$\|\boldsymbol{g} \star (\boldsymbol{f} + \boldsymbol{c})\| \approx \|\boldsymbol{g}\| \, \|\boldsymbol{f} + \boldsymbol{c}\| \, .$$

The fact that $\|\boldsymbol{f}\|, \|\boldsymbol{g}\|, \|\boldsymbol{c}\|$ are small thus implies that $\|\boldsymbol{h}\|$ is small[3]. The second condition is true because $\mathcal{F}_\Omega$ is a ring homomorphism.

To forge a signature, a third party would need to produce an $\boldsymbol{h}$ which is short, and which satisfies the required evaluations at points in Ω. It is conjectured that finding such an $\boldsymbol{h}$ is no easier than solving the associated CVP .

2.2 Transcript weaknesses in previous PASS protocols

The difficulty with this PASS prototype is that a transcript of signatures produced by a single signer on any set of messages leaks information about that signer's secret key. One way to see this is via a ring homomorphism $\rho : R_q \to R_q$ given by

$$\rho(a_0 + a_1\boldsymbol{x} + a_2\boldsymbol{x}^2 + \cdots + a_{N-1}\boldsymbol{x}^{N-1}) = a_0 + a_{N-1}\boldsymbol{x} + a_{N-2}\boldsymbol{x}^2 + \cdots + a_1\boldsymbol{x}^{N-1}.$$

The homomorphism ρ plays the same role that conjugation would play if x were replaced by a primitive N^{th} root of unity. If a polynomial $\boldsymbol{p} \in R_q$ is drawn randomly from a distribution, let $\mathbb{E}[\boldsymbol{p}]$ denote the expectation of $\boldsymbol{p}$, that is, the average of $\boldsymbol{p}$ over many samples. A third party observing many examples of $\boldsymbol{g} \star (\boldsymbol{f} + \boldsymbol{c})$ could compute

$$\mathbb{E}[\boldsymbol{g} \star (\boldsymbol{f} + \boldsymbol{c}) \star \rho(\boldsymbol{g} \star (\boldsymbol{f} + \boldsymbol{c}))] = \mathbb{E}[\boldsymbol{g} \star \rho(\boldsymbol{g})]\,\mathbb{E}[(\boldsymbol{f} + \boldsymbol{c}) \star \rho(\boldsymbol{f} + \boldsymbol{c})]$$

For simplicity assume that $\mathbb{E}[\boldsymbol{c}] = 0$, then, since $\boldsymbol{f}$ is constant, the above becomes

$$\mathbb{E}[\boldsymbol{g} \star \rho(\boldsymbol{g})]\left(\mathbb{E}[\boldsymbol{c} \star \rho(\boldsymbol{c})] + \boldsymbol{f} \star \rho(\boldsymbol{f})\right).$$

The distributions from which $\boldsymbol{c}$ and $\boldsymbol{g}$ are drawn are known, and thus a sufficiently long transcript will reveal $\boldsymbol{f} \star \rho(\boldsymbol{f})$ from which $\boldsymbol{f}$ may be computed by a technique from Gentry and Szydlo [8].

2.3 Recent developments and countermeasures

The problem with PASS was not that individual signatures leaked information about the secret key, but rather that an average over a collection of signatures would converge to a secret key dependent value. This is not a concern for signature schemes based on number theoretic trapdoor permutations, as such schemes enjoy relatively simple proofs that their signatures are uniformly distributed over the full range of possibilities. However, the requirement that PASS signatures have small norm, i.e. that they occupy a small region of the full domain, necessitates throwing out much of the algebraic structure that makes such uniformity

[3] The original PASS protocol used the centered L^2 norm - the L^2 norm about the mean of the vector. This norm can be seen to enjoy the above quasi-multiplicative property for independent random polynomials by considering the product in the complex Fourier domain, noting that the centering operation has the effect of zeroing the constant terms, and by applying Parseval's theorem.

guarantees possible. Full decoupling of secret keys from transcripts was a difficult barrier for the construction of secure lattice based signature schemes, and more so for the construction of efficient schemes.

The first successful decoupling, the signature scheme of Gentry, Peikert, and Vaikuntanathan [7], involved computing a candidate signature point $\boldsymbol{x}$ and then adding noise sampled from a discrete Gaussian distribution centered at $-\boldsymbol{x}$. The resulting signatures have a distribution which is computationally indistinguishable from a spherical discrete Gaussian centered at the origin.

Lyubashevsky, in [14], constructed a lattice based identification scheme which avoids transcript analysis attacks with a technique he called "aborting." In this scheme, provers are capable of determining when their response to a challenge will leak information about their secret key. Whenever this occurs they abort the protocol rather than supply a response.

In [15], Lyubashevsky improved his aborting technique and constructed a signature scheme through the Fiat-Shamir transform with hardness based on the Ring-SIS problem. Improvements and variants of this scheme with different hardness assumptions were presented in [16].

The first truly practical lattice signature scheme to avoid transcript attacks was developed by Güneysu, Lyubashevsky, and Pöppelmann [9]. Their scheme is a highly optimized variant of [16] and relies on a stronger hardness assumption.

The current state of the art would appear to be the new scheme, called BLISS, by Ducas, Durmus, Lepoint, and Lyubashevsky [4]. This scheme makes use of an NTRU-like key generation procedure and a bimodal discrete Gaussian noise distribution to produce very compact signatures. The efficiency of the scheme is also very impressive, especially considering the complexity of sampling discrete Gaussians.

3 PASS$_{\mathsf{RS}}$ – PASS with Rejection Sampling

N	-	Dimension
q	-	Prime $\equiv 1 \pmod{N}$
g	-	a primitive N^{th} root of unity in $\mathbb{Z}_q$
Ω	-	A subset of $\{g^j : 1 \leq j \leq N-1\}$
t	-	$\lvert\Omega\rvert$
k	-	Infinity norm of noise polynomials
b	-	1-norm of challenge polynomials

Table 1: Public parameters

We now present PASS$_{\mathsf{RS}}$ a new variant of PASS which completely decouples the transcript distribution from the secret key. Table 1 lists the public parameters of the system and gives a brief description of each.

Some notes on notation: R_q is the ring $\mathbb{Z}_q[x]/(x^N - 1)$; elements $\boldsymbol{a} \in R_q$ are represented as polynomials $\boldsymbol{a} = a_0 + a_1\boldsymbol{x} + a_2\boldsymbol{x}^2 + \cdots + a_{N-1}\boldsymbol{x}^{N-1}$, with coefficients in $a_i \in \mathbb{Z}_q$. We freely transition between this polynomial representation and a coefficient vector representation, $\boldsymbol{a} = [a_0, a_1, a_2, \ldots, a_{N-1}]^T$, wherever convenient.

Norms, such as $\|\boldsymbol{a}\|_\infty$ and $\|\boldsymbol{a}\|_1$, are the standard L^p norms on coefficient vectors; for numerical calculations we consistently identify a_i with an integer such that $|a_i| \leq q/2$.

We write $\mathcal{B}^1(b)$ to denote the elements of R_q with 1-norm $\leq b$, and $\mathcal{B}^\infty(k)$ to denote the elements of R_q with ∞-norm $\leq k$.

Lastly, The indicator function $\mathbf{1}_\mathcal{S}(x)$ yields 1 if $x \in \mathcal{S}$ and 0 otherwise.

3.1 Key Generation

A secret key is a polynomial with L^∞ norm equal to 1. We recommend the simple strategy of choosing each coefficient independently and uniformly from $\{-1, 0, 1\}$. Binary coefficients, though attractive for several reasons, would open the system up to a UniqueSVP gap amplification attack similar to that used by Nguyen in his cryptanalysis of GGH [19].

The public key corresponding to the secret key $\boldsymbol{f}$ is $\widehat{\boldsymbol{f}}|_\Omega = \mathcal{F}_\Omega \boldsymbol{f}$.

3.2 Signing

Signing is an iterated process consisting of the generation of a candidate signature followed by a rejection sampling step to prevent the publication of candidates that could leak secret key information.

A party with secret key $\boldsymbol{f}$, who wishes to sign a message μ, first selects a commitment polynomial $\boldsymbol{y}$ uniformly at random from $\mathcal{B}^\infty(k)$. The commitment $\boldsymbol{y}$ serves to mask the private key and must be treated with the same care as the private key itself. The signer then computes and stores $\widehat{\boldsymbol{y}}|_\Omega = \mathcal{F}_\Omega \boldsymbol{y}$, which will ultimately be made public if the candidate passes rejection sampling.[4]

Next, the signer computes a challenge, $\boldsymbol{c}$, which binds $\widehat{\boldsymbol{y}}|_\Omega$ to μ. To do so she makes use of the public algorithms:

$$\mathsf{Hash} : \mathbb{Z}_q^t \times \{0,1\}^* \to \{0,1\}^\ell, \text{and}$$

$$\mathsf{FormatC} : \{0,1\}^\ell \hookrightarrow \mathcal{B}^1(b).$$

Hash concatenates its inputs and passes the result through a cryptographic hash function such as SHA-512. FormatC maps the set of bitstrings output by Hash into a set of sparse polynomials. We avoid further description of the algorithms for now and simply say that

$$\boldsymbol{c} = \mathsf{FormatC}(\mathsf{Hash}(\widehat{\boldsymbol{y}}|_\Omega, \mu)).$$

[4] Note that the generation of $\boldsymbol{y}$ and the computation of $\widehat{\boldsymbol{y}}|_\Omega$ can both be done offline, oblivious to the message to be signed.

Finally, the signer computes a candidate signature point

$$\boldsymbol{z} = \boldsymbol{f} \star \boldsymbol{c} + \boldsymbol{y} \in R_q,$$

if any of the coefficients of $\boldsymbol{z}$ fall outside the interval $[-k+b, k-b]$, then $\boldsymbol{y}$, $\boldsymbol{c}$, and $\boldsymbol{z}$ are discarded and the signing process is repeated. Otherwise, the signer outputs the signature $(\boldsymbol{c}, \boldsymbol{z}, \mu)$.

In section 4 we will prove that signatures that pass the rejection sampling procedure have $\boldsymbol{z}$ values that are uniformly distributed over $\mathcal{B}^{\infty}(k-b)$.

3.3 Verification

The signature $(\boldsymbol{c}, \boldsymbol{z}, \mu)$ is valid if $\boldsymbol{z}$ is in $\mathcal{B}^{\infty}(k-b)$ and if

$$\boldsymbol{c} = \mathsf{FormatC}(\mathsf{Hash}(\widehat{\boldsymbol{z}}|_{\Omega} - \widehat{\boldsymbol{f}}|_{\Omega} \odot \widehat{\boldsymbol{c}}|_{\Omega}, \mu)).$$

Since $\mathcal{F}_{\Omega}$ is a ring homomorphism, it is the case that $\widehat{\boldsymbol{z}}|_{\Omega} = \widehat{\boldsymbol{f}}|_{\Omega} \odot \widehat{\boldsymbol{c}}|_{\Omega} + \widehat{\boldsymbol{y}}|_{\Omega}$. Therefore, on receipt of $(\boldsymbol{c}, \boldsymbol{z}, \mu)$, any verifier in possession of the appropriate public key $\widehat{\boldsymbol{f}}|_{\Omega}$ can evaluate $\boldsymbol{z}$ and $\boldsymbol{c}$ and compute $\widehat{\boldsymbol{y}}|_{\Omega} = \widehat{\boldsymbol{z}}|_{\Omega} - \widehat{\boldsymbol{f}}|_{\Omega} \odot \widehat{\boldsymbol{c}}|_{\Omega}$. The correctness of the scheme is immediate.

Algorithm 1 Sign

```
Input: $(\mu, \boldsymbol{f})$
repeat
    $\boldsymbol{y} \xleftarrow{\$} \mathcal{B}^{\infty}(k)$
    $h \leftarrow \mathsf{Hash}(\widehat{\boldsymbol{y}}|_{\Omega}, \mu)$
    $\boldsymbol{c} \leftarrow \mathsf{FormatC}(h)$
    $\boldsymbol{z} \leftarrow \boldsymbol{f} \star \boldsymbol{c} + \boldsymbol{y}$
until $\boldsymbol{z} \in \mathcal{B}^{\infty}(k-b)$
Output: $(\boldsymbol{c}, \boldsymbol{z}, \mu)$
```

Algorithm 2 Verify

```
Input: $(\boldsymbol{c}, \boldsymbol{z}, \mu, \widehat{\boldsymbol{f}}|_{\Omega})$
$result \leftarrow$ invalid
if $\boldsymbol{z} \in \mathcal{B}^{\infty}(k-b)$ then
    $h' \leftarrow \mathsf{Hash}(\widehat{\boldsymbol{z}}|_{\Omega} - \widehat{\boldsymbol{f}}|_{\Omega} \odot \widehat{\boldsymbol{c}}|_{\Omega}, \mu)$
    $\boldsymbol{c}' \leftarrow \mathsf{FormatC}(h')$
    if $\boldsymbol{c} = \boldsymbol{c}'$ then
        $result \leftarrow$ valid
    end if
end if
Output: $result$
```

4 Rejection Sampling

Each iteration of the signature generation routine produces a candidate signature which is accepted or rejected based on its infinity norm alone. In this section we will argue that this rejection sampling procedure completely decouples the distribution of signature points from the private key.

We will make use of the following fact:

Fact 1 *Each candidate signature $\boldsymbol{z}$ is in $\mathcal{B}^{\infty}(k+b)$.*

Proof. By definition we have $\|\boldsymbol{z}\|_{\infty} = \|\boldsymbol{f} \star \boldsymbol{c} + \boldsymbol{y}\|_{\infty}$ and by the triangle inequality: $\|\boldsymbol{f} \star \boldsymbol{c} + \boldsymbol{y}\|_{\infty} \leq \|\boldsymbol{f} \star \boldsymbol{c}\|_{\infty} + \|\boldsymbol{y}\|_{\infty}$. Again by the triangle inequality, $\|\boldsymbol{f} \star \boldsymbol{c}\|_{\infty} \leq \|\boldsymbol{f}\|_{\infty} \|\boldsymbol{c}\|_{1}$, thus

$$\|\boldsymbol{z}\|_{\infty} \leq \|\boldsymbol{f}\|_{\infty} \|\boldsymbol{c}\|_{1} + \|\boldsymbol{y}\|_{\infty} \leq b + k.$$

We will also make use of the following assumption on instantiations of Hash and $\mathsf{FormatC}$.

Assumption 1 *Let the public parameters* (N, q, k, b, Ω) *be fixed and let* $\boldsymbol{c} \in \mathcal{B}^1(b)$*,* $\boldsymbol{y} \in \mathcal{B}^\infty(k)$*,* $\mu \in \{0,1\}^*$ *be random variables related by*

$$\boldsymbol{c} = \mathsf{FormatC}(\mathsf{Hash}(\widehat{\boldsymbol{y}}|_\Omega, \mu)).$$

We assume that Hash *is a collision resistant hash function, that* $\boldsymbol{c}$ *and* $\boldsymbol{y}$ *are independent, and that* $\boldsymbol{c}$ *is uniform over the range of* $\mathsf{FormatC}$*. More explicitly, for any fixed* $\boldsymbol{c}_0 \in \mathcal{B}^1(b)$ *and fixed* $\boldsymbol{y}_0 \in \mathcal{B}^\infty(k)$*,*

$$\Pr[\boldsymbol{c} = \boldsymbol{c}_0 \mid \boldsymbol{y} = \boldsymbol{y}_0] = \frac{\Pr[\boldsymbol{c} = \boldsymbol{c}_0]\Pr[\boldsymbol{y} = \boldsymbol{y}_0]}{\Pr[\boldsymbol{y} = \boldsymbol{y}_0]} = |\mathcal{B}^1(b)|^{-1}.$$

Note that assumption 1 is no stronger than the standard random oracle assumption, so the reader may assume we are working in the random oracle model. We state the assumption in the above form to aid in the analysis of concrete instantiations. Clearly the assumption that the joint distribution of $\boldsymbol{y}$ and $\boldsymbol{c}$ factors is untenable - no deterministic instantiation of Hash can satisfy it while maintaining collision resistance. Yet by choosing an appropriate padding scheme for μ one should be able to approximately satisfy the assumption. We leave the exploration of padding schemes and analysis of the practical impact of assumption 1 to future work.

The following proposition describes the distribution of candidate signatures.

Proposition 1. *Fix vectors* $\boldsymbol{f}_0 \in \mathcal{B}^\infty(1)$ *and* $\boldsymbol{z}_0 \in \mathcal{B}^\infty(k+b)$*. Then as the pair* $(\boldsymbol{c}, \boldsymbol{y})$ *is chosen uniformly from the space* $\mathcal{B}^1(1) \times \mathcal{B}^\infty(k)$*, we have*

$$\Pr[\boldsymbol{f}_0 \star \boldsymbol{c} + \boldsymbol{y} = \boldsymbol{z}_0] = |\mathcal{B}^\infty(k)|^{-1} \sum_{\boldsymbol{c}_0 \in \mathcal{B}^1(b)} \Pr[\boldsymbol{c} = \boldsymbol{c}_0]\, \mathbf{1}_{\mathcal{B}^\infty(k)}(\boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0).$$

Proof. For any fixed $\boldsymbol{c}_0 \in \mathcal{B}^1(b)$ we have

$$\begin{aligned}\Pr[\boldsymbol{f}_0 \star \boldsymbol{c}_0 + \boldsymbol{y} = \boldsymbol{z}_0] &= \Pr[\boldsymbol{y} = \boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0] \\ &= \begin{cases} |\mathcal{B}^\infty(k)|^{-1} & \text{if } (\boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0) \in \mathcal{B}^\infty(k) \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

By application of the law of total probability and the assumption that the $\boldsymbol{c}$ and $\boldsymbol{y}$ are independent:

$$\begin{aligned}\Pr[\boldsymbol{f}_0 \star \boldsymbol{c} + \boldsymbol{y} = \boldsymbol{z}_0] &= \sum_{\boldsymbol{c}_0 \in \mathcal{B}^1(b)} \Pr[\boldsymbol{c} = \boldsymbol{c}_0]\Pr[\boldsymbol{f}_0 \star \boldsymbol{c} + \boldsymbol{y} = \boldsymbol{z}_0 \mid \boldsymbol{c} = \boldsymbol{c}_0] \\ &= \sum_{\boldsymbol{c}_0 \in \mathcal{B}^1(b)} \Pr[\boldsymbol{c} = \boldsymbol{c}_0]\Pr[\boldsymbol{y} = \boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0] \\ &= |\mathcal{B}^\infty(k)|^{-1} \sum_{\boldsymbol{c}_0 \in \mathcal{B}^1(b)} \Pr[\boldsymbol{c} = \boldsymbol{c}_0]\, \mathbf{1}_{\mathcal{B}^\infty(k)}(\boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0).\end{aligned}$$

Recall from section 3.2 that a candidate signature is rejected unless its $\boldsymbol{z}$ component is contained in $\mathcal{B}^\infty(k-b)$. The following proposition shows that each point in $\mathcal{B}^\infty(k-b)$ is selected as a candidate signature with equal probability.

Proposition 2. *Fix vectors $\boldsymbol{f}_0$ in $\mathcal{B}^\infty(1)$ and $\boldsymbol{z}_0$ in $\mathcal{B}^\infty(k-b)$. Then as the pair $(\boldsymbol{c}, \boldsymbol{y})$ is chosen uniformly from the space $\mathcal{B}^1(b) \times \mathcal{B}^\infty(k)$, we have*

$$\Pr\left[\boldsymbol{f}_0 \star \boldsymbol{c} + \boldsymbol{y} = \boldsymbol{z}_0\right] = |\mathcal{B}^\infty(k)|^{-1}.$$

Proof. We first note that $\mathcal{B}^\infty(k-b)$ is contained within $\mathcal{B}^\infty(k+b)$, so proposition 1 applies. Additionally, it is the case that $\|\boldsymbol{z}_0\|_\infty \le k-b$ and consequently, for any fixed $\boldsymbol{c}_0 \in \mathcal{B}^1(b)$, we have $\|\boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0\|_\infty \le k$. Thus $\boldsymbol{z}_0 - \boldsymbol{f}_0 \star \boldsymbol{c}_0$ is contained in $\mathcal{B}^\infty(k)$ and the indicator function in proposition 1 is unconditionally satisfied. Therefore,

$$\Pr\left[\boldsymbol{f}_0 \star \boldsymbol{c} + \boldsymbol{y} = \boldsymbol{z}_0\right] = |\mathcal{B}^\infty(k)|^{-1} \sum_{\boldsymbol{c}_0 \in \mathcal{B}^1(b)} \Pr\left[\boldsymbol{c} = \boldsymbol{c}_0\right] = |\mathcal{B}^\infty(k)|^{-1}.$$

Proposition 2 informs us that each of the $|\mathcal{B}^\infty(k-b)|$ acceptable signature points is chosen with probability $|\mathcal{B}^\infty(k)|^{-1}$. We infer that each pass through the signature generation routine has probability

$$\Pr\left[accept\right] = \frac{|\mathcal{B}^\infty(k-b)|}{|\mathcal{B}^\infty(k)|} = \left(1 - \frac{2b}{2k+1}\right)^N \approx e^{-\frac{Nb}{k}}$$

of generating a valid signature point, where the approximation is valid provided that both N and k/b are large.

A *transcript* is a set of signatures published by an honest signer. For instance, a signer who uses private key $\boldsymbol{f}$ to sign messages $\mu_1, \mu_2, \ldots, \mu_k$ produces a transcript

$$T = \left\{(\boldsymbol{c}_i, \boldsymbol{z}_i) : (\boldsymbol{c}_i, \boldsymbol{z}_i, \mu_i) = \mathsf{Sign}(\mu_i, \boldsymbol{f})\right\}.$$

Proposition 3. *A transcript T generated by an honest signer with private key $\boldsymbol{f}$ is indistinguishable from a set of points drawn uniformly from $\mathcal{B}^1(b) \times \mathcal{B}^\infty(k-b)$. Furthermore, for any fixed $\boldsymbol{c}_0 \in \mathcal{B}^1(b)$, $\boldsymbol{z}_0 \in \mathcal{B}^\infty(k-b)$ and $\boldsymbol{f}_0 \in \mathcal{B}^1(1)$, the events $(\boldsymbol{c}_0, \boldsymbol{z}_0) \in T$ and $\boldsymbol{f} = \boldsymbol{f}_0$ are independent.*

Proof. The $\boldsymbol{c}$ components of T are uniformly distributed over $\mathcal{B}^1(b)$ by assumption 1. Proposition 2 establishes not only that the $\boldsymbol{z}$ components of T are uniformly distributed over $\mathcal{B}^\infty(k-b)$, but also that the distribution of $\boldsymbol{z}$ depends *only* on the distribution of $\boldsymbol{y}$. Again by assumption 1, $\boldsymbol{c}$ and $\boldsymbol{y}$ are independent and therefore $\boldsymbol{c}$ and $\boldsymbol{z}$ are independent. The distribution of transcript points is consequently the product distribution of $\boldsymbol{c}$ and $\boldsymbol{z}$, i.e. uniform over $\mathcal{B}^1(b) \times \mathcal{B}^\infty(k-b)$.

Independence of transcript points from the secret key follows from the fact that proposition 2 holds for all choices of $\boldsymbol{f}_0$ in $\mathcal{B}^\infty(1)$.

5 Security Analysis

Our security analysis will focus on two types of attacks, those that target the hash function (or the combination $\mathsf{FormatC} \circ \mathsf{Hash}$), and those that target the partial Fourier transform $\mathcal{F}_\Omega$. Other attacks may be possible, and investigating them is an area for future work.

As our aim is to develop a practical quantum-resistant signature scheme, we will assume that the adversary has access to a quantum computer. Relatively little is known about the existence or non-existence of quantum algorithms for lattice problems, so our assumptions related to quantum computers will only address their ability to solve k-element black-box search problems in $\Theta(\sqrt{k})$ time.

5.1 Attacks on the hash function

The most obvious constraint on the security of the system comes from the entropy of $\boldsymbol{c}$. An adversary who can find a Hash preimage of a particular $\boldsymbol{c}$ can produce forgeries on structured messages from any user's public key. To do so, the adversary:

1. Chooses arbitrary $\boldsymbol{z}$ and $\boldsymbol{c}$ from the appropriate domains.
2. Computes $\widehat{\boldsymbol{g}}|_\Omega = \widehat{\boldsymbol{z}}|_\Omega - \widehat{\boldsymbol{f}}|_\Omega \odot \widehat{\boldsymbol{c}}|_\Omega$, where $\widehat{\boldsymbol{f}}|_\Omega$ is the victim's public key.
3. Finds a preimage of $\boldsymbol{c}$ in $\mathsf{Hash}(\widehat{\boldsymbol{g}}|_\Omega, \cdot)$.

While attacks against specific hash functions can have arbitrarily low complexity, we will assume that a strong hash function is chosen, and only consider generic attacks. If the output of Hash is r bits, a quantum adversary can find preimages in time $\Theta(2^{r/2})$. For κ-bit security, the range of $\mathsf{FormatC} \circ \mathsf{Hash}$ should produce an essentially uniform distribution on a set of cardinality $2^{2\kappa}$.

5.2 Attacks on the partial Fourier transform

An adversary who can find $\mathcal{F}_\Omega$ preimages which are in $\mathcal{B}^\infty(k-b)$ can forge signatures on arbitrary messages from any user's public key.

1. Adversary chooses random point $\boldsymbol{g}_F$ in $\mathcal{B}^\infty(k)$
2. $\boldsymbol{c}_F = \mathsf{FormatC}(\mathsf{Hash}(\mathcal{F}_\Omega \boldsymbol{g}_F, \mu))$
3. $\widehat{\boldsymbol{z}_F}|_\Omega = \widehat{\boldsymbol{g}_F}|_\Omega + \widehat{\boldsymbol{f}}|_\Omega \widehat{\boldsymbol{c}_F}|_\Omega$
4. Adversary uses preimage attack on $\widehat{\boldsymbol{z}_F}|_\Omega$ to find appropriate $\boldsymbol{z}_F$.

Adversaries could also try to recover the secret key directly with their preimage algorithm, but in order for this to be effective they must be able to find exceptionally short preimages. The problem of secret key recovery seems, at least intuitively then, to be harder than forgery. Yet, surprisingly, given the particular parameters of the scheme, lattice attacks may be better suited for solving the secret key recovery problem than they are for forging messages. Some care must be taken when choosing parameters to balance the difficulty of the two problems.

Lattice attacks on $\mathcal{F}_{\Omega}$ As mentioned briefly in the introduction, the partial Fourier recovery problem can easily be seen to be no harder than a specific class of closest vector problem CVP. Presented with the evaluation set, Ω, and a partial Fourier representation $\widehat{\boldsymbol{z}}|_{\Omega}$, an adversary can construct a lattice in which solving the CVP associated to any arbitrary preimage of $\widehat{\boldsymbol{z}}|_{\Omega}$ allows them to construct a short preimage of $\widehat{\boldsymbol{z}}|_{\Omega}$.

That lattice, which we denote $\Lambda^{\perp}(\mathcal{F}_{\Omega})$, is equivalent to the kernel of $\mathcal{F}_{\Omega}$,

$$\Lambda^{\perp}(\mathcal{F}_{\Omega}) = \left\{ \boldsymbol{a} \in \mathbb{Z}_q^N \;:\; \mathcal{F}_{\Omega}\boldsymbol{a} = \boldsymbol{0} \pmod{q} \right\}.$$

In practice, CVP instances are almost always solved by transforming the problem into an SVP in dimension $N+1$. If $\boldsymbol{z}'$ is an arbitrary preimage of the target $\widehat{\boldsymbol{z}}|_{\Omega}$, i.e. $\mathcal{F}_{\Omega}\boldsymbol{z}' = \widehat{\boldsymbol{z}}|_{\Omega}$ but $\|\boldsymbol{z}'\|$ is large, and $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \dots, \boldsymbol{b}_m\}$ form a Hermite Normal Form basis for $\Lambda^{\perp}(\mathcal{F}_{\Omega})$, then solving SVP in the lattice generated by the columns of

$$\mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{z}'} = \begin{pmatrix} q & & 0 & \boldsymbol{b}_{1,0} & \dots & \boldsymbol{b}_{m,0} & \boldsymbol{z}'_0 \\ & \ddots & & \vdots & \ddots & \vdots & \vdots \\ 0 & & q & \boldsymbol{b}_{1,t-1} & \dots & \boldsymbol{b}_{m,t-1} & \boldsymbol{z}'_{t-1} \\ 0 & \dots & 0 & \vdots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \vdots & \boldsymbol{b}_{1,N-1} & \dots & \boldsymbol{b}_{m,N-1} & \boldsymbol{z}'_{N-1} \\ 0 & \dots & 0 & 0 & \dots & 0 & 1 \end{pmatrix}$$

is likely to yield a short $\boldsymbol{z}$ such that $\mathcal{F}_{\Omega}\boldsymbol{z} = \widehat{\boldsymbol{z}}|_{\Omega}$.

Experiments by Micciancio and Regev [18] have demonstrated that lattice reduction algorithms perform best against the kernel lattices, $\Lambda^{\perp}(\boldsymbol{A})$, of $t \times N$ matrices $\boldsymbol{A}$ when $N \approx \sqrt{t \log(q)/\log(\gamma)}$ for some $\gamma \approx 1.01$ determined experimentally for each reduction algorithm. In the $\text{PASS}_{\mathsf{RS}}$ setting this places restrictions on t and q that we have obeyed in all of our proposed parameter sets. As such there should be no benefit to attacking a sublattice of $\mathcal{L}^{\mathsf{SVP}}$, and we proceed under this assumption.

The performance of lattice reduction algorithms, particularly LLL and BKZ, on lattices such as $\mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{z}'}$ is difficult to analyze in practice. Perhaps the most surprising complicating factor is that the performance depends crucially on the coset of $\mathbb{Z}_q^N/\Lambda^{\perp}(\mathcal{F}_{\Omega})$ to which $\boldsymbol{z}'$ belongs, and not strongly on $\boldsymbol{z}'$ itself. This dependence gives rise to two regimes that we will analyze separately. The extreme case, when $\boldsymbol{z}'$ is very close to the kernel lattice, produces instances of the $\mathsf{UniqueSVP}$ problem and determines the difficulty of the secret key recovery problem in $\text{PASS}_{\mathsf{RS}}$. The average case produces instances of $\mathsf{ApproxSVP}$ which will inform our discussion of the signature forgery problem.

$\mathsf{UniqueSVP}$ is the problem of finding a shortest vector in a lattice that is known to have a significant gap between the lengths of its first and second successive minima. Such is the case[5] in the lattices $\mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{f}'}$, as the the secret key, $\boldsymbol{f}$, has an expected norm of $\sqrt{2N/3}$ and $[\boldsymbol{f},\ 1]^T \in \mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{f}'}$.

[5] Curiously, the fact that the kernel lattice always contains the exceptionally short vector $[1, 1, \dots, 1]$ seems to have no impact here.

Lattice reduction algorithms can be ranked according to the so-called Hermite factor that they achieve. Algorithms that achieve Hermite factor γ can be expected to find the shortest vector in a lattice when the $\mathsf{UniqueSVP}$-gap, $\lambda_2(\mathcal{L})/\lambda_1(\mathcal{L})$, is greater than a constant fraction of γ. This behavior was first examined by Gama and Nguyen, whose experiments determined that for a certain class of random lattices the constant is approximately 0.48 [5]. They exhibited classes of lattices for which the constant was smaller, but these appear to be somewhat exceptional. Ducas et al. [4] performed similar experiments on the lattices that occur in BLISS, and found the constant again to be 0.48, and we have found the same to be true of the lattices related to $\mathrm{PASS}_{\mathsf{RS}}$.

Table 2 contains estimates on the Hermite factor needed to recover $\mathrm{PASS}_{\mathsf{RS}}$ secret keys at several concrete parameter levels. We estimate $\lambda_2(\mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{f}'})$ by the Gaussian heuristic in the L^2 norm. This predicts that N successive minima of a lattice will be tightly clustered around the radius of the smallest N-ball that has volume equal to the determinant of the lattice. The q-ary lattices, $\Lambda^{\perp}(\mathcal{F}_\Omega)$, have determinant q^t, and the Gaussian heuristic therefore predicts

$$\lambda_2(\mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{f}'}) = \lambda_1(\Lambda^{\perp}(\mathcal{F}_\Omega)) \approx \det(\Lambda^{\perp}(\mathcal{F}_\Omega))^{1/N}\sqrt{\tfrac{N}{2\pi\mathbf{e}}} = q^{t/N}\sqrt{\tfrac{N}{2\pi\mathbf{e}}}.$$

As mentioned above, we estimate λ_1 as $\sqrt{2N/3}$, the length of the secret key. This gives us a $\mathsf{UniqueSVP}$-gap, $\lambda_2/\lambda_1 \approx q^{t/N}\sqrt{3/(4\pi\mathbf{e})}$. Incorporating the constant 0.48 adjustment, we find that lattice reduction algorithms must achieve Hermite factor

$$\gamma = 0.62 \cdot q^{t/N} \tag{1}$$

in order to recover $\mathrm{PASS}_{\mathsf{RS}}$ secret keys.

The analysis for forgery attacks is very similar, only now the target $\widehat{\boldsymbol{z}}|_\Omega$ will lie in an essentially random coset of $\mathbb{Z}_q^N/\Lambda^{\perp}(\mathcal{F}_\Omega)$. The relevant problem is now $\mathsf{ApproxSVP}_\alpha$ the problem of finding a short vector that is more than α factor of being optimal, in other words a vector that is no longer than $\alpha\lambda_1(\mathcal{L}^{\mathsf{SVP}}_{\boldsymbol{z}'})$. Lattice reduction algorithms that achieve Hermite factor γ can solve $\mathsf{ApproxSVP}$ with factor $\alpha = \gamma^2$ in the worst case. That said, $\alpha = \gamma$ seems achievable on average [5], so we use this estimate in our analysis.

$\mathrm{PASS}_{\mathsf{RS}}$ signatures are validated by the L^∞ norm, but lattice reduction algorithms typically only guarantee the L^2 norm of their results. A vector of L^2 norm $\sqrt{N}\cdot(k-b)$ could potentially serve as a forgery, but this is highly unlikely. We estimate the approximation factor to be the ratio of the expected length of a forgery to the volume of the lattice, which is

$$\alpha = \sqrt{N}\cdot V/q^{t/N}, \tag{2}$$

where V is the variance of the discrete uniform distribution on $[-k+b, k-b]$.

Concrete performance of lattice reduction algorithms Current folklore is that lattice reduction algorithms can achieve Hermite factor $\approx 1.01^N$ in reasonable time but that Hermite factor 1.005^N is completely out of reach. These are useful heuristics, but they reflect more our ignorance about the concrete performance of lattice reduction and enumeration algorithms than they do our knowledge. Unfortunately, it seems that we know far too little about how these algorithms perform in high dimension to give precise "bit-security" estimates. We can, however, roughly determine which of the currently available lattice reduction algorithms might be useful for attacking $\text{PASS}_{\texttt{RS}}$.

Experiments by Schneider and Buchmann [21] indicate that the Hermite factor reachable by BKZ with blocksize β is approximately:

$$1.01655 - 0.000196185 \cdot \beta,$$

which for Hermite factors relevant to our parameter sets yields:

Blocksize (β)	15	30	40	55
Root Hermite factor	1.0136	1.0107	1.0087	1.0058

Table 2 lists several $\text{PASS}_{\texttt{RS}}$ parameter sets, the line labeled "Lattice security factor" represents our best guess as to the Hermite factor needed to launch either a key recovery or forgery attack (whichever is easier). We expect that our toy parameter set, $N = 433$, could be defeated by running BKZ-15 to completion. Although we do not have a good estimate on how long this would take, it should be possible with current technology.

Our other parameter sets should be significantly more difficult to attack. While Hermite factor 1.01^N is nominally within reach of today's technology, this has only been verified in relatively small dimensions. We know very little about how the algorithms will perform in dimension 577. Key recovery attacks on this parameter set should be possible with BKZ-30, but other approaches are likely needed to make the attack practical.

Chen and Nguyen have had impressive success with their BKZ-2.0 algorithm [3], which combines extreme pruning, developed in [6], with an early termination procedure, theoretically justified by [11]. BKZ-2.0 runs BKZ at phenomenally high blocksizes for a small number of rounds under the experimentally justified belief that most of the progress of BKZ is made in the early rounds. It is difficult to extrapolate security estimates from the results published thus far on BKZ-2.0's performance, but it would appear that our $577, 769$, and 1153 parameter sets could be within reach of terminated BKZ-75, 122, and 229 respectively.

For $N = 577$, our experiments with a BKZ-2.0 simulator similar to that presented in [3] indicate that 56 rounds of BKZ-75 would be sufficient to reach root hermite factor 1.0106; for $N = 769$, 47 rounds of BKZ-122 would suffice to reach 1.0084; and for $N = 1153$, 42 rounds of BKZ-229 would reach 1.0058.

Following the analysis of [3], we expecte enumeration to be the most expensive subroutine of BKZ-2.0. Each round consists of approximately N enumerations, and the cost of each enumeration depends on the the number of nodes visited in the enumeration tree. The estimated bit security is

$$\log_2(N \cdot rounds) + \log_2(nodes\ per\ enumeration) + \log_2(cost\ per\ node)$$

Using number-of-node and cost-per-node estimates from [3], we have that the estimated security of our $N = 769$ parameter is $\log_2(769 \cdot 47) + 53 + \log_2(200) \approx 76$ bits.

For $N = 1153$, a single enumeration in BKZ-229 is expected to take over 2^{130} time, which is greater than the expected time for a quantum attack on the hash function.

N	433	577	769	1153
q	775937	743177	1047379	968521
g	268673	296108	421722	56574
k	$2^{12}-1$	$2^{14}-1$	$2^{15}-1$	$2^{15}-1$
b	19	24	29	36
t	200	280	386	600
$\Pr[Accept]$	0.78	0.57	0.49	0.72
UniqueSVP gap	1.0117	1.0093	1.0075	1.0052
ApproxSVP factor	1.0105	1.0101	1.0081	1.0054
Lattice security factor	1.0134	1.0106	1.0084	1.0058
Entropy of $\boldsymbol{c}$	124	160	200	260
Bit-security bound	$\ll 62$	$\ll 80$	< 100	≤ 130

Table 2: Parameter sets and security indicators. UniqueSVP gap refers to λ_2/λ_1 without any correction for the performance of specific lattice reduction algorithms.

6 Reference Implementation

We have created a reference implementation of PASS_{RS} in C and made it available[6] under the GNU General Public License. Table 3 gives some idea of the performance of PASS_{RS} relative to the recent proposal of Ducas et al. (BLISS [4]) and to RSA and ECDSA. BLISS was tested using the June 13, 2013 version[7]. The implementations of RSA and ECDSA are from OpenSSL 1.0.1e. All benchmarks were run on a single 2.8GHz core of an Intel Core i7-2640M with hyper threading and turbo boost disabled. We make no claims as to the accuracy of these benchmarks - the timing methods used internally by the three libraries tested are incommensurate and many variables have been left uncontrolled. However, we do feel that these preliminary performance estimates are worth reporting, as they indicate that the schemes are competitive with each other and that further comparisons would be interesting.

[6] https://github.com/NTRUOpenSourceProject/ntru-crypto

[7] http://bliss.di.ens.fr/

6.1 Performance considerations

The two most computationally intensive parts of $\text{PASS}_{\mathtt{RS}}$ are the number theoretic transforms (NTT) used to compute $\mathcal{F}_\Omega$, and the sparse cyclic convolution used in computing $\boldsymbol{z} = \boldsymbol{f} \star \boldsymbol{c} + \boldsymbol{y}$. To compute $\mathcal{F}_\Omega$ we use Rader's algorithm to decompose the prime length NTT into cyclic convolution of length $N - 1$. We compute the resulting convolution as a pair of Fourier transforms over $\mathbb{C}$ using version 3.3.3 of FFTW. For all of the parameter sets presented above we have chosen chosen N to be a Pierpont prime (a prime of the form $2^u \cdot 3^v + 1$) as these yield very fast Fourier transform algorithms. Fermat primes $(2^u + 1)$ would yield a faster transforms, but there are no Fermat primes in our preferred parameter range.

We have made little effort to optimize the computation of sparse convolutions, and these often dominate the running time of the signing process.

6.2 Concrete instantiations of public functions

Our reference implementation uses SHA-512 to instantiate Hash for all parameter sets. The input passed to SHA-512 is the concatenation of the low order byte of each coefficient of $\widehat{\boldsymbol{y}}|_\Omega$ followed by the SHA-512 digest of μ.

$$\mathsf{Hash}(\widehat{\boldsymbol{y}}, \mu) = \text{SHA-512}(lowbyte(\widehat{\boldsymbol{y}}_0) \mid \ldots \mid lowbyte(\widehat{\boldsymbol{y}}_{t-1}) \mid \text{SHA-512}(\mu))$$

We have not implemented any message padding.

Our instantiation of $\mathsf{FormatC}$ sets aside the first 64 bits of $h_0 = \mathsf{Hash}(\widehat{\boldsymbol{y}}|_\Omega, \mu)$ to use as signs of the nonzero coefficients of $\boldsymbol{c}$. The remaining bits of h_0 are used, 16 at a time, in a rejection sampling procedure to generate uniform random values in the interval $[0, N-1]$. Each such value becomes the index of a non-zero coefficient of $\boldsymbol{c}$. If the pool of bits is ever exhausted, the process continues on $h_i = \text{SHA-512}(h_{i-1})$.

The random coefficients of $\boldsymbol{y}$ are generated by a rejection sampling procedure on the output of a stream cipher. Specifically we use the procedure from [10] of keying the Salsa20 stream cipher with a short seed from the Linux kernel random number generator.

Parameter Set	Sign		Verify	
	Median	Average	Median	Average
577	121996	171753	86828	87031
769	174900	205456	120204	120374
1153	421904	584230	172428	172641

Table 4: Sandy Bridge cycle counts for $\text{PASS}_{\mathtt{RS}}$. 100k samples

Algorithm	Parameter Set	Sign (μs)	Verify (μs)	Sig. (bytes)	Pub. key (bytes)
$\text{PASS}_{\texttt{RS}}$	577	62	31	1115	700
	769	73	40	1578	965
	1153	203	69	2360	1500
BLISS	0	321	25	413	413
	I	164	44	700	875
	II	642	43	625	875
	III	270	45	750	875
	IV	496	47	813	875
RSA	1024	225	15	128	128
	2048	1591	50	256	256
	4096	11532	185	512	512
ECDSA	`secp160r1`	80	270	40	20
	`nistp256`	146	348	64	32
	`nistp384`	268	1151	96	48

Table 3: Benchmarks. Times are averages over many operations.

References

1. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proceedings of the twenty-eighth annual ACM symposium on theory of computing. pp. 99–108. STOC 1996, ACM (1996)
2. Candes, E., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory 52(2), 489–509 (2006)
3. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: ASIACRYPT 2011, pp. 1–20. Springer (2011)
4. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, LNCS, vol. 8042, pp. 40–56. Springer (2013)
5. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N. (ed.) EUROCRYPT 2008, LNCS, vol. 4965, pp. 31–51. Springer (2008)
6. Gama, N., Nguyen, P.Q., Regev, O.: Lattice enumeration using extreme pruning. In: EUROCRYPT 2010. LNCS, vol. 6110, pp. 257–278. Springer (2010)
7. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th annual ACM symposium on theory of computing. pp. 197–206. STOC 2008, ACM (2008)
8. Gentry, C., Szydlo, M.: Cryptanalysis of the revised NTRU signature scheme. In: Knudsen, L.R. (ed.) EUROCRYPT 2002, LNCS, vol. 2332, pp. 299–320. Springer (2002)
9. Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Practical lattice-based cryptography: A signature scheme for embedded systems. In: Prouff, E., Schaumont, P. (eds.) CHES 2012, LNCS, vol. 7428, pp. 530–547. Springer (2012)
10. Güneysu, T., Oder, T., Pöppelmann, T., Schwabe, P.: Software speed records for lattice-based signatures. In: Post-Quantum Cryptography. LNCS, vol. 7932, pp. 67–82. Springer (2013)

11. Hanrot, G., Pujol, X., Stehlé, D.: Analyzing blockwise lattice algorithms using dynamical systems. In: Rogaway, P. (ed.) CRYPTO 2011, LNCS, vol. 6841, pp. 447–464. Springer (2011)
12. Hoffstein, J., Kaliski, B.S.J., Lieman, D.B., Robshaw, M.J.B., Yin, Y.L.: Secure user identification based on constrained polynomials (2000), U.S. Classification: 713/168; 380/28; 380/30; 713/170; 713/176 International Classification: H04L 932; H04L 928; H04L 930
13. Hoffstein, J., Silverman, J.H.: Polynomial rings and efficient public key authentication II. In: Lam, K.Y., Shparlinski, I., Wang, H., Xing, C. (eds.) Cryptography and Computational Number Theory, Progress in Computer Science and Applied Logic, vol. 20, pp. 269–286. Birkhäuser (2001)
14. Lyubashevsky, V.: Lattice-based identification schemes secure under active attacks. In: Cramer, R. (ed.) PKC 2008, LNCS, vol. 4939, pp. 162–179. Springer (2008)
15. Lyubashevsky, V.: Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In: ASIACRYPT 2009, pp. 598–616. Springer (2009)
16. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012, LNCS, vol. 7237, pp. 738–755. Springer (2012)
17. Lyubashevsky, V., Micciancio, D.: Generalized compact knapsacks are collision resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) Automata, Languages and Programming, LNCS, vol. 4052, pp. 144–155. Springer (2006)
18. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.) Post-Quantum Cryptography, pp. 147–191. Springer (2009)
19. Nguyen, P.: Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from CRYPTO97. In: Wiener, M. (ed.) CRYPTO 1999, LNCS, vol. 1666, pp. 288–304. Springer (1999)
20. Peikert, C., Rosen, A.: Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In: Halevi, S., Rabin, T. (eds.) Theory of Cryptography, LNCS, vol. 3876, pp. 145–166. Springer (2006)
21. Schneider, M., Buchmann, J.: Extended lattice reduction experiments using the BKZ algorithm. In: Sicherheit, LNI, vol. 170, pp. 241–252. Gesellschaft für Informatik (2010)