Des. Codes Cryptogr. (2015) 77:541–552
DOI 10.1007/s10623-015-0089-z

PASS-Encrypt: a public key cryptosystem based on partial evaluation of polynomials

Jeffrey Hoffstein[1] · **Joseph H. Silverman**[1]

Received: 20 August 2014 / Revised: 19 April 2015 / Accepted: 22 April 2015 /
Published online: 9 May 2015

Abstract A new hard problem in number theory, based on partial evaluation of certain classes of constrained polynomials, was introduced in Hoffstein et al. (Secure user identification based on constrained polynomials, 2000) and further refined in Hoffstein et al. (Polynomial rings and efficient public key authentication, 1999; Practical signatures from the partial Fourier recovery problem, 2013), Hoffstein and Silverman (Polynomial rings and efficient public key authentication. II, 2001) to create an efficient authentication and digital signature scheme called PASS. In this paper we present a public key cryptosystem called PASS-Encrypt that is based on the same underlying hard problem. We also provide an alternative description in terms of partial knowledge of discrete Fourier transforms.

Keywords Lattice-based cryptography · Public key cryptography · Discrete Fourier transform

Mathematics Subject Classification Primary: 11T71 · Secondary: 11H71, 94A60

1 Introduction

We present PASS-Encrypt, a public key cryptosystem whose security is based on the difficulty of reconstructing a small polynomial from some of its values, or equivalently, on the difficulty

Dedicated to the memory of Scott Vanstone.

This is one of several papers published in *Designs, Codes and Cryptography* comprising the "Special Issue on Cryptography, Codes, Designs and Finite Fields: In Memory of Scott A. Vanstone".

✉ Joseph H. Silverman
jhs@math.brown.edu

Jeffrey Hoffstein
jhoff@math.brown.edu

[1] Mathematics Department, Brown University, Box 1917, Providence, RI 02912, USA

of reconstructing a short vector from partial knowledge of its discrete Fourier transfrom. PASS-Encrypt is a PKC companion to the digital signature scheme called PASS (*Polynomial Authentication and Signature Scheme*) that was introduced in [11] and further refined in [7, 9]. We give theoretical descriptions of two versions of PASS-Encrypt and explain how the underlying partial DFT problem is equivalent to finding short or close vectors in certain lattices. Items that we do not discuss in this paper include (1) the use of standard methods to turn our basic scheme into a system that has both semantic and chosen-ciphertext security; (2) the possibility of creating provably secure variants of PASS-Encrypt. Thus one may view this note as proposing a candidate trapdoor one-way function, rather than describing a fully-fledged encryption scheme.

2 Some historical remarks

The mid-1990s saw a flurry of activity around the use of hard lattice problems to create public key cryptosystems and digital signature schemes that were either extremely efficient [6,8,9] or had desirable theoretical properties [1]. Among the former, the NTRUEncrypt PKC [8] developed by the authors and Jill Pipher was presented at the Crypto 96 rump session and published in 1998. NTRUEncrypt relies directly on SVP and CVP in certain cyclic modular lattices whose special structure accounts for its efficiency. Various signature schemes based on NTRU-type lattices have been proposed over the years, but until recently [13] they all suffered from transcript information leakage. (See also [2,5,17,18] for other work on amelioration of transcript leakage in NTRU signatures.) The mid-1990s also saw the introduction by the authors and Dan Lieman [7,9] of a digital signature scheme called PASS that was based on the difficulty of reconstructing a small polynomial from a partial list of its values. PASS, which was published in 1999, also suffered from transcript leakage, although a recent variant [12] uses rejection sampling to eliminate this problem. Shortly after PASS was originally developed, the authors realized that the PASS problem could also be used as the basis for a public key cryptosystem, and they wrote up some preliminary notes (dated January 14, 1999). However, since it was clear at that time that NTRUEncrypt would be more efficient and have smaller key and ciphertext sizes, they did not further develop PASS-Encrypt.

The past decade has seen a resurgence of interest in lattice-based cryptosystems for a variety of reasons, including their potential resistance to quantum algorithms, their use by Gentry in the construction of a fully homomorphic encryption scheme [4], and other work such as the construction of multilinear pairings [3]. In view of this activity and the recent introduction of a transcript secure PASS signature scheme [12], the authors felt that it might be of interest to publish their practical and efficient PKC based on the PASS problem, despite the fact that PASS-Encrypt remains somewhat less efficient than NTRUEncrypt, and despite the fact that we have not investigated variants of PASS-Encrypt that might admit formal proofs of security.

3 Discrete Fourier transformations and polynomial evaluation

We set the following notation, which will remain fixed throughout this paper.

$\mathbb{F}_q$ a finite field with q elements.
N a prime satisfying $q \equiv 1 \pmod{N}$.
p a small prime that does not divide q.

w a primitive N^{th} root of unity in $\mathbb{F}_q$.
S an ordered subset of $\{0, 1, 2, \ldots, N-1\}$. Let $s = \#S$.
S' the ordered complement of S, i.e., $S' = \{0 \le i \le N-1 : i \notin S\}$. Let $s' = \#S'$.

For $\boldsymbol{a} \in \mathbb{F}_q^N$, the *Discrete Fourier Transformation of* $\boldsymbol{a}$ is the vector $\mathcal{F}(\boldsymbol{a}) \in \mathbb{F}_q^N$ whose k^{th} coordinate is

$$\mathcal{F}(\boldsymbol{a})_k = \sum_{i=0}^{N-1} a_i w^{ik}.$$

The *Inverse Discrete Fourier Transform* $\mathcal{F}^{-1}$ is defined by the similar formula

$$\mathcal{F}^{-1}(\boldsymbol{a})_k = \frac{1}{N} \sum_{i=0}^{N-1} a_i w^{-ik}.$$

It is easy to verify that

$$\mathcal{F}^{-1} \circ \mathcal{F}(\boldsymbol{a}) = \boldsymbol{a} \qquad \text{and} \qquad \mathcal{F} \circ \mathcal{F}^{-1}(\boldsymbol{a}) = \boldsymbol{a}.$$

Thus knowledge of a vector is equivalent to knowledge of its Fourier transform, albeit subject to a certain amount of computation.

For any subset $S = \{k_1, k_2, \ldots, k_s\}$ of indices, we let $\mathcal{F}_S(\boldsymbol{a})$ be the corresponding subset of the coordinates of $\mathcal{F}(\boldsymbol{a})$:

$$\mathcal{F}_S(\boldsymbol{a}) = \big(\mathcal{F}(\boldsymbol{a})_{k_1}, \mathcal{F}(\boldsymbol{a})_{k_2}, \ldots, \mathcal{F}(\boldsymbol{a})_{k_s}\big).$$

We call $\mathcal{F}_S(\boldsymbol{a})$ a *Partial Fourier Transform*, in the sense that it contains partial information about the full Fourier transform. Note that knowledge of a partial Fourier transform is not, in general, sufficient to determine the original vector. Indeed, if $s \ne N$, then there will be many vectors with the same S-partial Fourier transform.

There are two types of products that are useful. The first is simply component-by-component multiplication,

$$\boldsymbol{a} \cdot \boldsymbol{b} = (a_0 b_0, b_1 b_1, \ldots, a_{N-1} b_{N-1}).$$

The second is the convolution product $a \otimes b$ whose k^{th} coordinate is given by the formula

$$(\boldsymbol{a} \otimes \boldsymbol{b})_k = \sum_{i=0}^{N-1} a_i b_{k-i},$$

where it is understood that $b_j = b_{j+N}$ if $j < 0$.

It is a standard, and very important, fact that the Fourier transform is a homomorphism from the ring of vectors with convolution product to the ring of vectors with component multiplication. That is, it satisfies

$$\mathcal{F}(\boldsymbol{a} + \boldsymbol{b}) = \mathcal{F}(\boldsymbol{a}) + \mathcal{F}(\boldsymbol{b}) \qquad \text{and} \qquad \mathcal{F}(\boldsymbol{a} \otimes \boldsymbol{b}) = \mathcal{F}(\boldsymbol{a}) \cdot \mathcal{F}(\boldsymbol{b}).$$

Remark 1 Fourier transformations and vector multiplications can also be described in terms of polynomials. To do this, we identify the vector $\boldsymbol{a}$ with the polynomial

$$a(X) = a_0 + a_1 X + a_2 X^2 + \cdots + a_{N-1} X^{N-1}.$$

Then the k^{th} Fourier coefficient of $\boldsymbol{a}$ is simply the value of $a(X)$ at $X = w^k$,

$$\mathcal{F}(\boldsymbol{a})_k = a(w^k).$$

Similarly, the inverse Fourier transform is $\mathcal{F}^{-1}(\boldsymbol{a})_k = N^{-1}a(w^{-k})$. Notice that the partial Fourier transformation $\mathcal{F}_S(\boldsymbol{a})$ is the list of values $a(w^k)$ for the indices $k \in S$.

Continuing with the polynomial interpretation, we note that the convolution product $\boldsymbol{a}\otimes\boldsymbol{b}$ is equal to the product $a(X)b(X)$ in the quotient ring $\mathbb{F}_q[X]/(X^N-1)$. In other words, the polynomial corresponding to $\boldsymbol{a}\otimes\boldsymbol{b}$ is equal to the polynomial obtained by first multiplying $a(X)b(X)$ as polynomials and then setting $X^{N+i} = X^i$ for $0 \le i < N$.

4 PASS-Encrypt: version 1

Public parameters: The quantities q, N, w, S, and S' are public cryptosystem parameters that are chosen based on the desired speed and security level.
Key creation: Bob's private key is a small vector $\boldsymbol{f} \in \mathbb{F}_q^N$. Bob's public key is the partial Fourier Transform $\mathcal{F}_S(\boldsymbol{f})$ of f. For decryption purposes, Bob will probably want to compute and store (privately) the full Fourier Transform $\mathcal{F}(\boldsymbol{f})$.
Encryption: Alice's plaintext[1] is a moderately small vector $\boldsymbol{m} \in \mathbb{F}_q^N$, where the actual information transmitted is the value of $\boldsymbol{m}$ modulo p. She also chooses a small random vector $\boldsymbol{r} \in \mathbb{F}_q^N$. She computes $\mathcal{F}(\boldsymbol{m})$ and $\mathcal{F}(\boldsymbol{r})$. Using these values and Bob's public key $\mathcal{F}_S(\boldsymbol{f})$, she computes the following three quantities:

$$\begin{aligned}\boldsymbol{e} &= p\mathcal{F}_S(\boldsymbol{f})\cdot\mathcal{F}_S(\boldsymbol{r}) + \mathcal{F}_S(\boldsymbol{m}).\\ \boldsymbol{e}' &= \mathcal{F}_{S'}(\boldsymbol{r}).\\ \boldsymbol{e}'' &= \mathcal{F}_{S'}(\boldsymbol{m}).\end{aligned}$$

She transmits to Bob the ciphertext $(\boldsymbol{e}, \boldsymbol{e}', \boldsymbol{e}'')$. Note that since $\boldsymbol{e}'$ and $\boldsymbol{e}''$ are only partial Fourier Transforms of $\boldsymbol{r}$ and $\boldsymbol{m}$, they do not allow an attacker to directly reconstruct $\boldsymbol{r}$ and $\boldsymbol{m}$.
Decryption: Bob combines his private knowledge of $\mathcal{F}_{S'}(\boldsymbol{f})$ with the values $\boldsymbol{e}'$ and $\boldsymbol{e}''$ that he received from Alice to compute

$$p\mathcal{F}_{S'}(\boldsymbol{f})\cdot\mathcal{F}_{S'}(\boldsymbol{r}) + \mathcal{F}_{S'}(\boldsymbol{m}).$$

Combining this with $\boldsymbol{e}$, Bob knows the value of

$$p\mathcal{F}(\boldsymbol{f})\cdot\mathcal{F}(\boldsymbol{r}) + \mathcal{F}(\boldsymbol{m}).$$

Taking the inverse Fourier transform of this quantity yields

$$p\boldsymbol{f}\otimes\boldsymbol{r} + \boldsymbol{m}.$$

Since the coefficients of $\boldsymbol{f}$ and $\boldsymbol{r}$ are small relative to q, this vector may be reduced modulo p to yield the message $\boldsymbol{m}$.

Table 1 illustrates the information that is public and the information that is private. An attacker knows only the four boxes marked Public, which is insufficient information to recover the message. Bob additionally knows the Private Key box, and using that information he is able to reconstruct the information that belongs in the empty box. He then knows the complete last column, i.e., he knows the complete discrete Fourier transformation of $p\boldsymbol{f}\otimes\boldsymbol{r} + \boldsymbol{m}$, so he is able to recover $p\boldsymbol{f}\otimes\boldsymbol{r} + \boldsymbol{m}$ itself, and then by reduction modulo p he recovers $\boldsymbol{m}$.

[1] In practice, of course, the actual plaintext must be suitably padded to achieve chosen-ciphertext security. Further, minor modifications are needed to achieve semantic security.

Table 1 The PASS-Encrypt public key cryptosystem

	f	m	r	$pf\otimes r + m$
S	Bob's Public Key	Alice's Private Message	Alice's Private Value	Alice's Public Message
S'	Bob's Private Key	Alice's Public Message	Alice's Public Value	

5 Homomorphic properties

In view of the current widespread interest in homomorphic encryption, we make some brief remarks on homomorphic properties of PASS-Encrypt. Addition works in a straightforward way. Alice computes the sums

$$(\boldsymbol{e}_1, \boldsymbol{e}'_1, \boldsymbol{e}''_1) + (\boldsymbol{e}_2, \boldsymbol{e}'_2, \boldsymbol{e}''_2)$$

and returns them to Bob. Bob can then reconstruct the full Fourier transform

$$\begin{aligned} &p\mathcal{F}(\boldsymbol{f}) \cdot \mathcal{F}(\boldsymbol{r}_1) + \mathcal{F}(\boldsymbol{m}_1) + p\mathcal{F}(\boldsymbol{f}) \cdot \mathcal{F}(\boldsymbol{r}_2) + \mathcal{F}(\boldsymbol{m}_2) \\ &\quad = \mathcal{F}\big(p\boldsymbol{f}\otimes(\boldsymbol{r}_1 + \boldsymbol{r}_2) + (\boldsymbol{m}_1 + \boldsymbol{m}_2)\big), \end{aligned}$$

and hence he can recover

$$p\boldsymbol{f}\otimes(\boldsymbol{r}_1 + \boldsymbol{r}_2) + (\boldsymbol{m}_1 + \boldsymbol{m}_2).$$

Assuming that the coefficients are not too large, i.e., have magnitude at most $q/2$, reduction modulo p yields the plaintext sum $\boldsymbol{m}_1 + \boldsymbol{m}_2$.

The situation with products is less straightforward due to the presence of cross terms. In order to recover $\boldsymbol{m}_1\otimes\boldsymbol{m}_2$ from a computation involving the encryptions of $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$, Bob needs to know the following four quantities:

$$\boldsymbol{e}_1 \cdot \boldsymbol{e}_2, \qquad \boldsymbol{e}'_1 \cdot \boldsymbol{e}'_2, \qquad \boldsymbol{e}''_1 \cdot \boldsymbol{e}''_2, \qquad \boldsymbol{e}'_1 \cdot \boldsymbol{e}''_2 + \boldsymbol{e}'_2 \cdot \boldsymbol{e}''_1. \tag{1}$$

To make it clear why Bob needs the fourth quantity, we set the notation that if $\boldsymbol{v} = \mathcal{F}_{S'}(\boldsymbol{u})$ is the S'-partial DFT of the vector $\boldsymbol{u}$, then $\mathcal{C}(\boldsymbol{v}) = \mathcal{F}_S(\boldsymbol{u})$ is the complementary S-partial DFT of $\boldsymbol{u}$. In particular, knowledge of both $\boldsymbol{v}$ and $\mathcal{C}(\boldsymbol{v})$ is enough to recover $\boldsymbol{v}$ via the inverse DFT.

With this notation, we see that

$$\begin{aligned} \boldsymbol{e}_1 \cdot \boldsymbol{e}_2 &= \Big(p\mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{F}_S(\boldsymbol{r}_1) + \mathcal{F}_S(\boldsymbol{m}_1)\Big) \cdot \Big(p\mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{F}_S(\boldsymbol{r}_2) + \mathcal{F}_S(\boldsymbol{m}_2)\Big) \\ &= \Big(p\mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{C}(\boldsymbol{e}'_1) + \mathcal{C}(\boldsymbol{e}''_1)\Big) \cdot \Big(p\mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{C}(\boldsymbol{e}'_2) + \mathcal{C}(\boldsymbol{e}''_2)\Big) \\ &= p^2\mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{C}(\boldsymbol{e}'_1) \cdot \mathcal{C}(\boldsymbol{e}'_2) \\ &\quad + p\mathcal{F}_S(\boldsymbol{f}) \cdot \Big(\mathcal{C}(\boldsymbol{e}'_1) \cdot \mathcal{C}(\boldsymbol{e}''_2) + \mathcal{C}(\boldsymbol{e}'_2) \cdot \mathcal{C}(\boldsymbol{e}''_1)\Big) + \mathcal{C}(\boldsymbol{e}''_1) \cdot \mathcal{C}(\boldsymbol{e}''_2). \end{aligned}$$

Hence using the four quantities (1) and his knowledge of the full DFT of $\boldsymbol{f}$, Bob can recover the full Fourier transform

$$\Big(p\mathcal{F}(\boldsymbol{f})\cdot\mathcal{F}(\boldsymbol{r}_1)+\mathcal{F}(\boldsymbol{m}_1)\Big)\cdot\Big(p\mathcal{F}(\boldsymbol{f})\cdot\mathcal{F}(\boldsymbol{r}_2)+\mathcal{F}(\boldsymbol{m}_2)\Big)$$
$$=\mathcal{F}\Big((p\boldsymbol{f}\otimes\boldsymbol{r}_1+\boldsymbol{m}_1)\otimes(p\boldsymbol{f}\otimes\boldsymbol{r}_2+\boldsymbol{m}_2)\Big).$$

Applying the inverse DFT gives

$$(p\boldsymbol{f}\otimes\boldsymbol{r}_1+\boldsymbol{m}_1)\otimes(p\boldsymbol{f}\otimes\boldsymbol{r}_2+\boldsymbol{m}_2),$$

and again assuming that the coefficients are not too large, reduction modulo p gives the value of $\boldsymbol{m}_1\otimes\boldsymbol{m}_2 \mod p$.

In general, to decrypt an n-fold product requires $n+2$ polynomial combinations of the original encrypted values. The first of these is simply $\boldsymbol{e}_1\cdot\boldsymbol{e}_2\cdots\boldsymbol{e}_n$. For the others, we obtain one for each $0\le k\le n$ by computing

$$\sum_{\substack{I\subset\{1,2,\ldots,n\}\\ \#I=k}}\prod_{i\in I}\boldsymbol{e}'_i\cdot\prod_{i\notin I}\boldsymbol{e}''_i \quad \text{(the product is coordinate-wise).}$$

These values and the knowledge of $\boldsymbol{f}$ allow Bob to recover the full Fourier transform of

$$\prod_{i=1}^{n}(p\boldsymbol{f}\otimes\boldsymbol{r}_i+\boldsymbol{m}_i) \quad \text{(the product is convolution),}$$

and then reduction modulo p recovers $\prod\boldsymbol{m}_i \mod p$ if the coefficients are not too large.

6 PASS-Encrypt: version 2

At the cost of some additional computation, it is possible to improve various operating characteristics of PASS-Encrypt, including key sizes and bandwidth. The basic idea is to switch the p factor from the product $\boldsymbol{f}\otimes\boldsymbol{r}$ to the plaintext $\boldsymbol{m}$. The cryptosystem parameters and the public and private keys are the same in both versions of PASS-Encrypt, so a single key can be used.

For concreteness, we make the following choices:

- $p=3$
- $\boldsymbol{f}$ and $\boldsymbol{r}$ are chosen as random vectors with coordinates equal to -1, 0, or 1.
- $\boldsymbol{m}$ is chosen with coordinates in the range $-M/2<m_i\le M/2$. An appropriate choice for M is $M\approx q/6$.

Public Parameters: As in Version 1, the quantities q, N, w, S, and S' are public cryptosystem parameters as before.
Key Creation: As in Version 1, Bob's private key is a small vector $\boldsymbol{f}\in\mathbb{F}_q^N$ and Bob's public key is the partial Fourier Transform $\mathcal{F}_S(\boldsymbol{f})$ of f.
Encryption: Alice's plaintext is a vector $\boldsymbol{m}\in\mathbb{F}_q^N$ with coordinates $-M/2<m_i\le M/2$. She also chooses a small random vector $\boldsymbol{r}\in\mathbb{F}_q^N$ as in Version 1. She computes $\mathcal{F}(\boldsymbol{m})$ and $\mathcal{F}(\boldsymbol{r})$. Using these values and Bob's public key $\mathcal{F}_S(\boldsymbol{f})$, she computes the following three quantities:

$$\boldsymbol{e} = \mathcal{F}_S(\boldsymbol{f}) \cdot \mathcal{F}_S(\boldsymbol{r}) + p\mathcal{F}_S(\boldsymbol{m}).$$
$$\boldsymbol{e}' = \mathcal{F}_{S'}(\boldsymbol{r}).$$
$$\boldsymbol{e}'' = \mathcal{F}_{S'}(\boldsymbol{m}).$$

She transmits to Bob the encrypted message $(\boldsymbol{e}, \boldsymbol{e}', \boldsymbol{e}'')$.

Decryption: Bob combines his private knowledge of $\mathcal{F}_{S'}(\boldsymbol{f})$ with the values $\boldsymbol{e}'$ and $\boldsymbol{e}''$ that he received from Alice to compute

$$\mathcal{F}_{S'}(\boldsymbol{f}) \cdot \mathcal{F}_{S'}(\boldsymbol{r}) + p\mathcal{F}_{S'}(\boldsymbol{m}).$$

Combining this with $\boldsymbol{e}$, Bob knows the value of

$$\mathcal{F}(\boldsymbol{f}) \cdot \mathcal{F}(\boldsymbol{r}) + p\mathcal{F}(\boldsymbol{m}).$$

Taking the inverse Fourier transform of this quantity yields

$$\boldsymbol{f} \otimes \boldsymbol{r} + p\boldsymbol{m}.$$

For appropriate choices of parameters, the coordinates of this vector will be in the range from $-q/2$ to $q/2$, so reduction modulo p yields the value of the product $\boldsymbol{f} \otimes \boldsymbol{r} \mod p$. Since Bob knows $\boldsymbol{f}$, he can multiply by $\boldsymbol{f}^{-1} \mod p$ to recover $\boldsymbol{r} \mod p$. But $\boldsymbol{r}$ has coordinates 0 and ± 1, so its value modulo p (with $p = 3$) determines $\boldsymbol{r}$ exactly. Finally, Bob has all of the information necessary to compute

$$\frac{(\boldsymbol{f} \otimes \boldsymbol{r} + p\boldsymbol{m}) - \boldsymbol{f} \otimes \boldsymbol{r}}{p} = \boldsymbol{m}$$

in $\mathbb{F}_q^N$, and since the coordinates of m satisfy $|m_i| \leq M < q/2$, this identifies m exactly.

7 Security analysis of lattice reduction attacks

In this section we reformulate the PASS-Encrypt problem as a lattice problem and briefly discuss its practical security against lattice reduction algorithms. See [12, Sect. 5.2.1] for further discussion.

7.1 Reduction of PASS-Encrypt to a lattice problem

In this section we describe how to recover a PASS-Encrypt key by searching for a small vector in a lattice. We leave for the reader the analogous reduction of the plaintext recovery problem to a closest vector problem. We also note that one may formulate the key recovery problem as a CVP, but in practice CVP is solved by converting it to an SVP, which is what we do explicitly.

Recall that the public key is the partial Fourier transfrom $\mathcal{F}_S(\boldsymbol{f})$, or equivalently, some of the values of the polynomial $\boldsymbol{f}$. Treating the coordinates of $\boldsymbol{f}$ as unknowns, the attacker knows that these coordinates satisfy the following system of linear congruences modulo q:

$$\sum_{i=0}^{N-1} f_i w^{ik} \equiv A_k \pmod{q} \quad \text{for } k \in S.$$

Here w and the A_k's are known quantities. The first step is to solve the s congruences for s of the f_i's in terms of the others. Note that this is done over the finite field $\mathbb{F}_q$, so is elementary linear algebra. This yields:

$$f_{N-s+j} \equiv \sum_{i=0}^{N-s-1} B_{ij} f_i + C_j \pmod{q}, \quad 0 \le j < s. \tag{2}$$

Here the B_{ij}'s and the C_j's are known quantities, and the attacker is looking for a solution where the f_i's are small. To do this, he forms the lattice L generated by the rows of the following $(N+1)$-by-$(N+1)$ matrix, where to ease notation, we let $M = N - s - 1$:

$$\left(\begin{array}{cccc|c|ccc} 1 & 0 & \cdots & 0 & 0 & B_{00} & B_{01} & \cdots & B_{0,s-1} \\ 0 & 1 & \cdots & 0 & 0 & B_{10} & B_{11} & \cdots & B_{1,s-1} \\ \vdots & \vdots & \ddots & \vdots & 0 & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & B_{M0} & B_{M1} & \cdots & B_{M,s-1} \\ \hline 0 & 0 & \cdots & 0 & 1 & C_0 & C_1 & \cdots & C_{s-1} \\ \hline 0 & 0 & \cdots & 0 & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & q \end{array}\right) \tag{3}$$

The small target vector in L is created by multipying the top rows of the matrix by $f_0, f_1, \ldots, f_M$, adding them to the middle row (i.e., the row with the C_j's), and then adding appropriate multiples of the bottom rows (i.e., the q-rows) so that the congruences (2) give the other f_i's. In this way we see that the following vector is in L:

$$\boldsymbol{v} = (f_0, f_1, \ldots, f_M, 1, f_{M+1}, \ldots, f_{N-1}).$$

For an appropriate choice of PASS-Encrypt parameters (with $p = 3$), the coordinates of the vector $\boldsymbol{f}$ consist of an approximately equal number of 0's, 1's, and -1's, so the length τ of the target vector is

$$\tau = \|\boldsymbol{v}\| \approx \sqrt{2N/3}.$$

The lattice L has dimension $N+1$ and discriminant q^s, so the Gaussian heuristic [10, Sect. 6.5.3] says that the smallest expected nonzero vector in L should have length approximately

$$\sigma = \sqrt{\frac{\dim L}{2\pi e}} \cdot (\operatorname{Disc} L)^{1/\dim L} = \sqrt{\frac{N+1}{2\pi e}} \cdot q^{s/(N+1)}.$$

Typical PASS-Encrypt parameters sets have $s \approx N/2$ with $q = 2N+1$ and N large, so

$$\sigma \approx N/\sqrt{\pi e}.$$

This leads to a target-to-Gaussian ratio

$$\frac{\tau}{\sigma} \approx \sqrt{\frac{2\pi e}{3N}} = O\left(\frac{1}{\sqrt{\dim L}}\right).$$

Many experiments using varied implementations of lattice reduction algorithms have shown that such lattice problems tend to be exponentially hard as a function of the dimension; see for example [7–9,12,14].

Remark 2 The lattice generated by the rows of (3) has some resemblence to the standard NTRU lattice [8], but with an important difference. Ignoring the middle column and row, the upper-right "B-block" in an an NTRU lattice has a cyclic structure; each row of an NTRU

B-block is a barrel-cyclic shift of the previous row. This means that the NTRU lattice may be viewed as a rank-2 R-module for the ring $R = \mathbb{Z}[X]/(X^N - 1)$, or even as an ideal in a quadratic extension of R. But the B-block of the PASS lattice does not appear to have any sort of cyclic structure, and thus it is not an ideal (nor a module of low rank) for a ring of the form $\mathbb{Z}[X]/(P(X))$. However, at the cost of increasing the lattice dimension, we can search for a PASS key in a lattice that does have a cyclic structure, although it is a vanderMonde-type cyclicity, rather than the shift cyclicity of the NTRU lattice. Thus let $S = \{k_1, k_2, \ldots, k_s\}$ and consider the row-span of the $(N + s)$-by-$(N + s)$ matrix

$$\left(\begin{array}{cccc|c|cccc}
1 & 0 & \cdots & 0 & 0 & 1 & 1 & \cdots & 1 \\
0 & 1 & \cdots & 0 & 0 & w^{k_1} & w^{k_2} & \cdots & w^{k_s} \\
\vdots & \vdots & \ddots & \vdots & 0 & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 & 0 & w^{(N-1)k_1} & w^{(N-1)k_2} & \cdots & w^{(N-1)k_s} \\
\hline
0 & 0 & \cdots & 0 & 1 & -A_{k_1} & -A_{k_2} & \cdots & -A_{k_s} \\
\hline
0 & 0 & \cdots & 0 & 0 & q & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & 0 & q & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & 0 & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & q
\end{array}\right)$$

This lattice contains the short vector $(f_0, f_1, \ldots, f_N, 1, 0, \ldots, 0)$.

Remark 3 Alexander May [15] has suggested an improved lattice attack on the NTRU cryptosystem, and these ideas were further extended in [16]. However, virtually all of the gain in May's idea comes from the fact that the convolution modular lattices used for NTRU have a large number of target vectors that are related to one another by rotational symmetries of their coordinates. It is easily seen that PASS-Encrypt lattices do not have this property. With very high probability, there is exactly one valid target vector in the lattice L; that is, exactly one vector that can be used to decrypt a message. (More precisely, it is possible that a small multiple of $\boldsymbol{v}$, say $2\boldsymbol{v}$ or $3\boldsymbol{v}$, will also decrypt, so what we should really say is that there is a unique one-dimensional target subspace of L.) Hence the ideas described in [15,16] are not helpful for PASS-Encrypt lattices.

8 Sample parameters

We list some sample parameter sets in Table 2. For these sets, we assume that S and S' have approximately $N/2$ elements, and that the coordinates of the vectors $\boldsymbol{f}$ and $\boldsymbol{r}$ are chosen randomly uniformly from the set $\{-1, 0, 1\}$. Thus $\boldsymbol{f}$ and $\boldsymbol{r}$ are chosen from a sample space of size 3^N, so a collision search requires $3^{N/2}$ comparisons. Even for the smallest value of N in Table 2, the bit security of a collision search is $3^{N/2} = 3^{359/2} \approx 2^{284.5}$.

The public key consists of roughly $\frac{1}{2}N$ numbers modulo q, as do each of the three components of the ciphertext. So we have roughly

$$\text{Key Size} \approx \left\lceil \frac{N}{2} \log_2(q) \right\rceil \text{ bits}, \qquad \text{Ciphertext} \approx \left\lceil \frac{3N}{2} \log_2(q) \right\rceil \text{ bits},$$

The operating characteristics of PASS-Encrypt depend primarily on the speed of computing N-dimensional mod q discrete Fourier transforms and of generating N-dimensional random vectors (e.g., via SHA). When done naively, we note that a DFT requires $O(N^2)$ multiplications modulo q, but fast Fourier transforms reduce this to $O(N \log N)$, in which

Table 2 Sample PASS-Encrypt parameters

N	p	q	w	Public key	Ciphertext
359	3	719	121	1704 bits	5111 bits
641	3	1283	4	3310 bits	9928 bits
911	3	1823	25	4935 bits	14,803 bits
1223	3	2447	25	6884 bits	20,651 bits

Table 3 Operations used by PASS-Encrypt

	DFT	Random	$\otimes$	$\cdot$	$+$
Key creation	1	1	0	0	0
Encrypt	2	1	0	$\frac{1}{2}$	$\frac{1}{2}$
Decrypt (v. 1)	1	0	0	$\frac{1}{2}$	$\frac{1}{2}$
Decrypt (v. 2)	1	0	2	$\frac{1}{2}$	$\frac{1}{2}$

Table 4 Estimated PASS-Encrypt operating characteristics with comparisons to other systems

Algorithm	Parameter set	Encrypt (μs)	Decrypt (μs)	Ciphertext (bytes)	Public key (bytes)
PASS	641	34	66	1241	414
	911	51	121	1851	617
	1223	75	231	2582	861
RSA	1024	225	15	128	128
	2048	1591	50	256	256
	4096	11,532	185	512	512
ECDSA	`secp160r1`	80	270	40	20
	`nistp256`	146	348	64	32
	`nistp384`	268	1151	96	48

case a DFT becomes comparable to component-wise addition or multiplication. (Similarly for convolution products.) Table 3 lists the number of N-dimensional DFTs, random vectors, convolution multiplications, and component-wise additions and multiplications required for each aspect of PASS-Encrypt.

Since actual operating speeds may vary by an order of magnitude or more depending on the implementation of the basic operations, and since neither of the authors is skilled at programming optimizations, we use [12, Sect. 6] to make rough speed comparisons. The implementation of PASS signatures in [12] uses $N \in \{577, 769, 1153\}$, but takes values of q that are much larger than $2N + 1$, which tends to make the underlying lattice problem easier, while making operations modulo q more time consuming. Doing a rough interpolation/extrapolation using [12, Table 3], we obtain the estimates given in Table 4.

We performed experiments on PASS-Encrypt lattices with N ranging from 101 to 181 using a standard implementation of the LLL-BKZ algorithm and fit the output to a regression line of the form

$$\log(\text{Running Time}) = a \dim(L) + b.$$

Table 5 Extrapolated LLL-BKZ lattice breaking time for PASS-Encrypt

N	q	Time (MIPS-years)
359	719	2.2×10^5
641	1283	3.3×10^{14}
911	1823	2.1×10^{23}
1223	2447	3.0×10^{33}

The correlation coefficient was 0.979, so the linearity of the log-running time is quite good. Extrapolating to higher dimensional lattices gives the estimated running times in Table 5. For comparison, we note that the estimated times to break RSA-1024 and RSA-2048 are, respectively, 3×10^{11} MIPS-years and 3×10^{28} MIPS-years.

Acknowledgments The authors would like to thank the referees for their helpful comments and corrections. This research was partially supported by NSF EAGER DMS-1349908.

References

1. Ajtai M., Dwork C.: A public-key cryptosystem with worst-case/average-case equivalence. In: STOC '97 (El Paso, TX), pp. 284–293 (electronic). ACM, New York (1999).
2. Ducas L., Nguyen P.Q.: Faster Gaussian lattice sampling using lazy floating-point arithmetic. In: Advances in Cryptology–ASIACRYPT 2012. Lecture Notes in Computer Science, vol. 7658, pp. 415–432. Springer, Heidelberg (2012).
3. Garg S., Gentry C., Halevi S.: Candidate multilinear maps from ideal lattices. In: Advances in Cryptology–EUROCRYPT 2013. Lecture Notes in Computer Science, vol. 7881, pp. 1–17. Springer, Heidelberg (2013).
4. Gentry C.: Fully homomorphic encryption using ideal lattices. In: STOC'09—Proceedings of the 2009 ACM International Symposium on Theory of Computing, pp. 169–178. ACM, New York (2009).
5. Gentry C., Peikert C., Vaikuntanathan V.: Trapdoors for hard lattices and new cryptographic constructions [extended abstract]. In: STOC'08, pp. 197–206. ACM, New York (2008).
6. Goldreich O., Goldwasser S., Halevi S.: Public-key cryptosystems from lattice reduction problems. In: Advances in Cryptology–CRYPTO '97 (Santa Barbara, CA, 1997). Lecture Notes in Computer Science, vol. 1294, pp. 112–131. Springer, Berlin (1997).
7. Hoffstein J., Silverman J.H.: Polynomial rings and efficient public key authentication. II. In: Cryptography and Computational Number Theory (Singapore, 1999). Programe on Computer Science Applications Logic, vol. 20, pp. 269–286. Birkhäuser, Basel (2001).
8. Hoffstein J., Pipher J., Silverman J.H.: NTRU: a ring-based public key cryptosystem. In: Algorithmic Number Theory (Portland, OR, 1998). Lecture Notes in Computer Science, vol. 1423, pp. 267–288. Springer, Berlin (1998).
9. Hoffstein J., Lieman D., Silverman J.H.: Polynomial rings and efficient public key authentication. In: Cryptographic Techniques and e-Commerce (CryTEC'99). City University of Hong Kong Press, Hong Kong (1999).
10. Hoffstein J., Pipher J., Silverman J.H.: An Introduction to Mathematical Cryptography. Undergraduate Texts in Mathematics. Springer, New York (2008).
11. Hoffstein J., Kaliski B., Lieman D., Robshaw M., Yin Y.: Secure user identification based on constrained polynomials. United States Patent 6,076,163. Filed October 20, 1997, Issued June 13 (2000).
12. Hoffstein J., Pipher J., Schanck J., Silverman J.H., Whyte W.: Practical signatures from the partial Fourier recovery problem. Cryptology ePrint Archive, Report 2013/757 (2013). http://eprint.iacr.org/. Proceedings of ACNS 2014 (to appear).
13. Hoffstein J., Pipher J., Schanck J., Silverman J.H., Whyte W.: Transcript secure signatures based on modular lattices. In: PQCrypto 2014. Lecture Notes in Computer Science, vol. 8772, pp. 142–159. Springer, Switzerland (2014). To appear http://eprint.iacr.org/2014/457.
14. Howgrave-Graham N., Silverman J H., Whyte W.: Choosing parameter sets for NTRUEncrypt with NAEP and SVES-3. In: Topics in Cryptology–CT-RSA 2005. Lecture Notes in Computer Science, vol. 3376, pp. 118–135. Springer, Berlin (2005).

15. May A.: Auf polynomgleichungen basierende public-key-kryptosysteme, June 4, (1999). Johann Wolfgange Goethe-Universitat, Frankfurt am Main, Fachbereich Informatik. (Masters Thesis in Computer Science).
16. May A., Silverman J.H.: Dimension reduction methods for convolution modular lattices. In: Cryptography and Lattices (Providence, RI, 2001). Lecture Notes in Computer Science, vol. 2146, pp. 110–125. Springer, Berlin (2001).
17. Melchor C.A., Boyen X., Deneuville J.-C., Gaborit P.: Sealing the leak on classical ntru signatures. In: PQCrypto 2014. Lecture Notes in Computer Science, vol. 8772, pp. 1–21. Springer, Switzerland (2014). To appear http://eprint.iacr.org/2014/484.
18. Stehl D., Steinfeld R.: Making ntruencrypt and ntrusign as secure as standard worst-case problems over ideal lattices. Cryptology ePrint Archive, Report 2013/004 (2013) http://eprint.iacr.org/.