Cryptogr. Commun. (2012) 4:131–144
DOI 10.1007/s12095-011-0061-3

Fault analysis of the NTRUSign digital signature scheme

Abdel Alim Kamal · Amr M. Youssef

Received: 11 December 2010 / Accepted: 14 December 2011 / Published online: 6 January 2012

Abstract We present a fault analysis of the NTRUSign digital signature scheme. The utilized fault model is the one in which the attacker is assumed to be able to fault a small number of coefficients in a specific polynomial during the signing process but cannot control the exact location of the injected transient faults. For NTRUsign with parameters $(N,\ q = p^l,\ \mathcal{B},$ standard, $\mathcal{N})$, when the attacker is able to skip the norm-bound signature checking step, our attack needs one fault, succeeds with probability $\approx 1 - \frac{1}{p}$ and requires $O((qN)^t)$ steps when the number of faulted polynomial coefficients is upper bounded by t. The attack is also applicable to NTRUSign utilizing the transpose NTRU lattice but it requires double the number of fault injections. Different countermeasures against the proposed attack are investigated.

Keywords Side channel attacks · Lattice-based public key cryptosystems · Fault analysis and countermeasures · Digital signature schemes · NTRU

Mathematics Subject Classification (2010) 94A60

1 Introduction

NTRUSign [1–4] is a parameterized family of lattice-based public key digital signature schemes that is currently under consideration for standardization by the IEEE P1363 working group. It was proposed after the original NTRU signature scheme (NSS) [5] was broken [6, 7]. Similar to the GGH cryptosystem [8], the security of

A. A. Kamal · A. M. Youssef (✉)
Concordia Institute for Information Systems Engineering, Concordia University, Montreal, QC, H3G 1M8, Canada
e-mail: youssef@ciise.concordia.ca

A. A. Kamal
e-mail: a_kamala@ciise.concordia.ca

NTRUSign is related to the hardness of approximating the closest vector problem (CVP) in a lattice. However, NTRUSign uses a compact lattice, referred to as the NTRU lattices, instead of the GGH lattices [3]. This NTRU lattice has the property that a private $2n$-dimensional basis for the lattice can be described with 2 vectors, each with n coefficients, and a public basis can be described with a single n-dimensional vector. This enables public keys to be represented in $O(n \log n)$ space, rather than $O(n^2)$ as is the case with GGH signature schemes. Furthermore, operations take $O(n^2)$ time, as opposed to $O(n^3)$ for elliptic curve based cryptosystems and RSA private key operations.

1.1 Previous attacks on NTRUSign

During the past few years, several attempts for analyzing the security of NTRUSign have been presented. Min et al. [9] showed that the original version of NTRUSign signature scheme [2] which was proposed at CT-RSA'03 is not secure in terms of strongly existential forgeable, i.e., it is malleable. The proposed attack allows an adversary to forge new signatures for a message of her choice, given a signature for this message. This forgery requires a specific polynomial with small coefficient satisfying its norm value equal to zero. Even if this forgery does not admit an adversary to change the message, this attack limits the applications of this version of NTRUSign, which does not use perturbation, in several applications.

Szydlo [10] introduced a new lattice reduction technique applicable to the class of hypercubic lattices which arise during transcript analysis of certain NTRUSign signature schemes. After a few thousand signatures, key recovery amounts to discovering a hidden unitary matrix from its Gram matrix [10]. This case of the Gram matrix factorization problem is equivalent to finding the shortest vectors in the hypercubic lattice defined by a quadratic form. This work on reduction of hypercubic lattices shows that the transcript attacks are relevant to the security of the NTRUSign schemes. The practical security threat to the schemes, however, was not clear given the required large number of oracle calls. Furthermore, the attack does not seem to be applicable to NTRUSign with perturbation.

Nguyen and Regev [11] presented the first successful key-recovery attack on NTRUSign-251 without perturbation [1, 2]. Experimentally, this attack requires about 400 signatures to recover the secret key of this non-perturbed version of NTRUSign. The main idea of the attack is based on the following learning problem: Given many random points uniformly distributed over an unknown n-dimensional parallelepiped, recover the parallelepiped or an approximation thereof. Nguyen and Regev transformed this problem into a multivariate optimization problem which they solved by a gradient descent. Again, this attack does not seem to be directly applicable to NTRUSign with perturbation where, in this case, the attacker has to solve an extension of the hidden parallelepiped problem in which the parallelepiped is replaced by the Minkowski sum of two hidden parallelepipeds where the lattice spanned by one of the parallelepipeds is public but the other one is not.

1.2 Fault analysis attacks

Cryptanalytic attacks can be divided into two classes: pure mathematical attacks and side channel attacks. Pure mathematical attacks, similar to the ones surveyed

above, are traditional cryptanalytic techniques that rely only on known or chosen input-output pairs of the cryptographic function, and exploit the inner structure of the cipher to reveal secret key information or allow for a signature forgery in case of digital signature schemes. On the other hand, in side channel attacks, it is assumed that the attacker has some access to the cryptographic device, for example, by being able to make measurements with respect to time or power consumption.

Fault analysis is an example of side channel attacks in which the attacker is assumed to be able to induce faults in the cryptographic device and observe the faulty output. Then, by careful inspection of the faulty output, the attacker recovers the secret information, such as secret inner state or secret key. Fault attacks were first introduced by Boneh et al. [12] where they described attacks that targeted the RSA public key cryptosystem by exploiting a faulty Chinese Remainder Theorem (CRT) computation to factor the public modulus. Subsequently, fault analysis attacks were extended to other digital signature schemes (e.g., [13–15]). Other variants of the fault analysis attacks against the RSA cryptosystem were also investigated. The perturbation of public elements was considered as a real threat when Seifert presented an attack on the RSA signature checking mechanism [16, 17]. Then, Brier et al. [18] extended this work to full recovery of the private signing exponent for various RSA implementations. Berzati et al. [19, 20] address the issue of modifying the modulus during the exponentiation. Fault analysis attacks were also extended to symmetric systems such as DES [21] and later to AES [22]. Fault attacks against stream ciphers were introduced by Hoch et al. [23]. A fault analysis attack against the NTRU encryption algorithm, NTRUEncrypt, was presented in [24].

Different assumptions are usually made regarding both timing and location of injected faults [25]. The number of required faults in fault attacks varies depending on the assumed fault analysis model. In particular, some strong fault models assume that the adversary has precise full control on both timing and location of injected faults which implies that the attacker has full control on the location of faulted bits as well as the attacked operation. The fault model utilized in this work is slightly more relaxed. In particular, we assume that the attacker is able to fault a small number of coefficients in a specific polynomial during the signing process but cannot control the exact location of the injected transient faults.

Many ideas have been proposed to efficiently secure CRT based RSA signature computations against fault attacks. In [26], Shamir used a redundant way to compute the arguments used in the CRT computation of the RSA signature and checked their correctness before RSA combination. Shamir's countermeasure has two drawbacks. It requires the secret exponent which is not needed for the CRT computation of the signature and its error detection technique is based on decisional tests that should be avoided since they can be bypassed by higher order fault attacks which were introduced by Kim and Quisquater in [27] where they were able to practically break the first-order countermeasures for the RSA cryptosystem. The main idea is based on inducing the first fault during one of the exponentiation and then inducing a second fault to skip the error checking routine in the RSA-CRT scheme. Yen et al. [28] noted that fault injection on status register flags can bypass conditional checks in countermeasures. Hence, they introduced the concept of infective computation which aims at infecting the resulting signature, i.e., rendering it unusable for the attacker in the case where a fault is injected in one of the two exponentiations. Several subsequent countermeasures employed this infective computation approach [14, 25, 29, 30].

1.3 Motivation

One main advantage of the NTRU family of cryptosystems (i.e., both NTRUEncrypt and NTRUSign) is that NTRU is smaller and faster than other public key cryptosystems. According to the performance benchmarks, published on the NTRU company website (http://www.ntru.com/security-lab/), NTRU is about 5 to 200 times faster than RSA and ECC with similar security parameters. A recent comparison between hardware implementations of different digital signature schemes [31] shows that the power consumption of NTRUSign is significantly less than any other digital signature scheme published so far. These small footprints make NTRUsign ideal for handheld mobile devices, sensors, RFIDs, smartcards, and other resource constrained devices with limited computing resources. On the other hand, the typical deployment environment for these resource constrained devices makes it particularly susceptible to various forms of side channel attacks including fault analysis which has become a serious threat after cheap and low-tech methods of applying faults were presented [32]. It should also be noted that the limited resources and the severe lightweight requirements make the implementation of various countermeasures against these attacks a more challenging task compared to other environments.

The rest of the paper is organized as follows. In the next section, we introduce some of the basic definitions and notations used throughout the rest of the paper. We also review the relevant details of the NTRUSign algorithm. Our proposed attack, its success probability and complexity are presented in Section 3. Different countermeasures against fault analysis attacks of NTRUSign are presented in Section 4. Finally, our conclusion is given in Section 5.

2 Preliminaries

In this section, we briefly review some of the basic definitions and notations used throughout this paper. The key underlying structure of NTRUSign is the polynomial ring

$$R = \frac{\mathbb{Z}[x]}{x^N - 1},$$

where N is a prime integer. The signatures and the messages use a polynomial ring

$$R_q = \frac{\mathbb{Z}/q\mathbb{Z}[x]}{x^N - 1},$$

where the coefficients are taken mod q.

The basic operations in NTRUSign are addition and convolution multiplication. The multiplication of two polynomials $a(x), b(x) \in R$ is given by

$$a(x) \star b(x) = c(x), \tag{1}$$

where

$$c_k = \sum_{i=0}^{k} a_i b_{k-i} + \sum_{i=k+1}^{N-1} a_i b_{N+k-i} = \sum_{j=k-i \pmod N} a_i b_j.$$

Algorithm 1 NTRUSign with perturbation [1, 2, 33, 34]

Key Generation

1: INPUT: Integers(N), q, $\mathcal{N}$, $\mathcal{B} \geq 0$, (df, dg for binary polynomials), (d for trinary polynomials), and the NTRU lattice type = "standard" or "transpose". In the above set of parameters, N denotes the dimension of the polynomial ring used, i.e., polynomials are up to degree $N-1$. $\mathcal{N}$ denotes the norm-bound.

2: Generate $\mathcal{B}$ private lattice bases and one public lattice basis: Set i=$\mathcal{B}$. While $i \geq 0$ do:

   a. Randomly choose binary polynomials $f, g \in R$ with (df, dg) ones. For the parameter sets defined in [33, 34], choose trinary polynomials f, g with ($d+1$) +1s and d -1s. f needs to be invertible in R_q in case of using NTRU lattice type = "standard" and g needs to be invertible in R_q in case of using NTRU lattice type = "transpose".
   b. Find small polynomials $F, G \in R$ such that $f \star G - F \star g = q$.
   c. Set $f_i = f$. If NTRU lattice type = "standard", set $f'_i = F_i$ and $h_i = f_i^{-1} \star g_i \bmod q$. If NTRU lattice type = "transpose", set $f'_i = g_i$ and $h_i = f_i^{-1} \star F_i \bmod q$. Set i=i-1.

3: PUBLIC OUTPUT: The input parameters and the public key $h = h_0$.

4: PRIVATE OUTPUT: The set of polynomial $\{f_i, f'_i, h_i\}$ for i=0..$\mathcal{B}$.

Signing

1: INPUT: A digital document $D \in \mathcal{D}$ (a digital document space) and the private key set $\{f_i, f'_i, h_i\}$ for $i = 0 \cdots \mathcal{B}$.

2: Set r=0.

3: Set s=0, i=0. Encode r as a bit string. Set m_0=$H(D \parallel r)$, where "$\parallel$" denotes the concatenation. Set m=m_0.

4: Perturb the point using the private lattices: While $i \geq 1$:

   1. Calculate $A = \lfloor \frac{-f'_i \star m}{q} \rceil$, $B = \lfloor \frac{f_i \star m}{q} \rceil$, $s_i = A \star f_i + B \star f'_i \bmod q$.
   2. Set $m = s_i \star h_i - h_{i-1} \bmod q$.
   3. Set $s = s + s_i$. Set $i = i - 1$.

5: Sign the perturbed point using the lattice public key:
Set $A = \lfloor \frac{-f'_0 \star m}{q} \rceil$, $B = \lfloor \frac{f_0 \star m}{q} \rceil$, $s_0 = A \star f_0 + B \star f'_0 \bmod q$, $s = s + s_0$.

6: Check the signature:

   1. Set $b = \|s, s \star h - m_0 \bmod q\|$.
   2. if $b \geq \mathcal{N}$. Set $r = r + 1$ and go to step 3.

7: OUTPUT: The triplet (D, r, s).

Verifying

1: INPUT: A signed document(D, r, s) and the public key h

2: Encode r as a bit string. Set m=$H(D \parallel r)$.

3: Set $b = \|s, s \star h - m_0 \bmod q\|$.

4: OUTPUT: "valid" if $b < \mathcal{N}$, "invalid" otherwise.

For a real number r, $\lfloor r \rceil$ denotes the closest integer to r. From the NTRUSign specifications [4], $\lfloor r \rceil$ is evaluated as $\lfloor r \rceil = \lfloor r + 0.5 \rfloor$ where $\lfloor \cdot \rfloor$ denotes the floor function of the enclosed argument. Similarly, if a is a polynomial with real coefficients, $\lfloor a \rceil$ denotes that this operation is applied to each coefficient in the polynomial a. Let $a(x) = \sum_{i=0}^{N-1} a_i x^i$ be a polynomial in R. The centered norm of $a(x)$ is defined as the non-negative real number satisfying $\|a(x)\|^2 = \sum_{i=0}^{N-1} (a_i - \mu_a)^2 = \sum_{i=0}^{N-1} a_i^2 - \frac{1}{N}(\sum_{i=0}^{N-1} a_i)^2$, where $\mu_a = (1/N) \sum_{i=0}^{N-1} a_i$ is the average of the coefficients of $a(x)$. The centered norm in a direct sum module R^n is defined as the nonnegative real number satisfying $\|(a_0, a_1)\|^2 = \|a_0\|^2 + \|a_1\|^2$.

The original variant of the NTRUSign algorithm was proposed in [1]. A perturbation technique was later introduced in order to enhance the security of this scheme, particularly, against transcript attacks. Algorithm 1 shows this enhanced version of NTRUSign [2–4]. The original scheme can be seen as a special case of Algorithm 1 with $\mathcal{B} = 0$.

3 Proposed fault analysis of NTRUSign

To simplify our discussion, we first consider the NTRUSign with the "standard" NTRU lattice [2]. Steps 1–4 of the signing procedure in Algorithm 1 serve only to add perturbation δ to the original message m. Step 5 performs the actual signature operation, using the original NTRUSign scheme, on $m' = m + \delta$. Since δ is designed to be small enough, a valid signature on m' will be a valid signature on m. Our attack utilizes the fact that an attacker only needs to recover f_0, and consequently calculate g_0, in order to forge a signature that would pass the verification step in Algorithm 1. In other words, signatures generated by the setting $\mathcal{B} = 0$ in the signing algorithm, i.e., with no perturbation, will pass the verification algorithm even if the verifier assumes that it was generated with $\mathcal{B} > 0$. In fact, it will pass the verification step even if different sets of F and G are utilized as long as they satisfy the condition identified in step 2b in the key generation procedure of Algorithm 1.

The fault model in which we analyze NTRUSign is the one in which the attacker is assumed to be able to inject a transient fault in a small number of coefficients of the polynomial A or B in the signing algorithm (see step 6 in Fig. 1) but cannot control the exact location of injected faults, i.e., cannot specify which polynomial coefficients are to be corrupted by the induced faults.

If the fault injection process grantees that number of faulted polynomial coefficients is upper bounded by t, then the resulting faulty signature is given by

$$\widehat{s} = s_1 + \widehat{s_0},$$

where

$$\widehat{s_0} = \widehat{A} \star f_0 + B \star f_0' \bmod q,$$

$\widehat{A} = A + \epsilon \bmod q$, and $\epsilon = \sum_{j=1}^{t} \epsilon_{i_j} x^{i_j} \in R_q, 0 \le i_j < N$, denotes the resulting error polynomial with t non zero coefficients in the locations corresponding to the injected

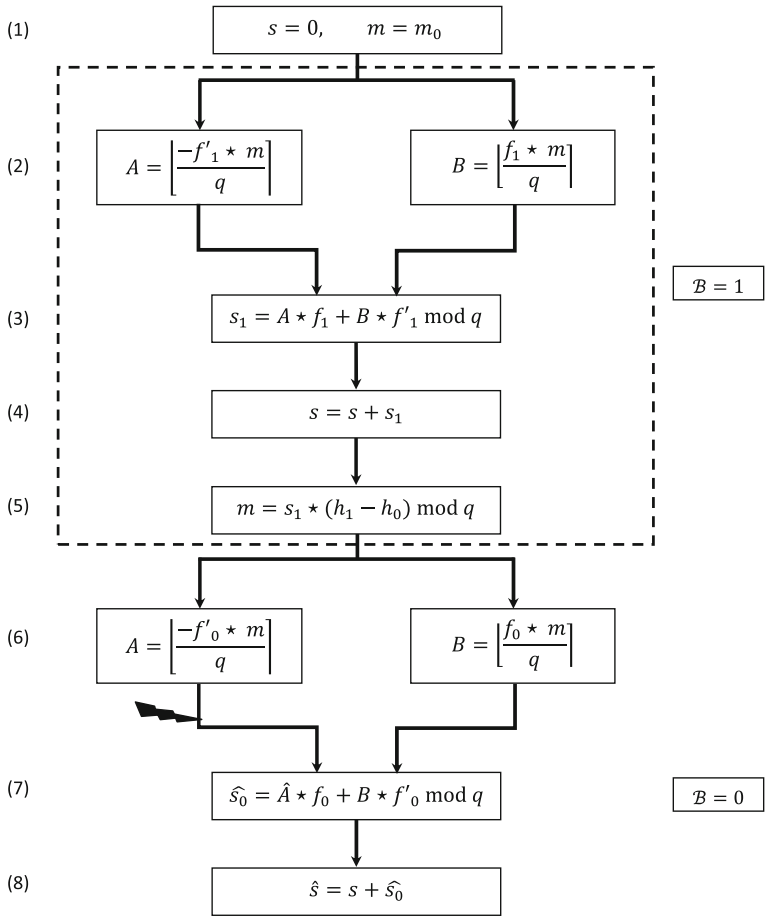

Fig. 1 Fault injection

faults. Thus, the attacker can calculate the difference between the faulty signature and the correct one as follows:

$$\begin{aligned} \Delta s = \widehat{s} - s &= (s_1 + \widehat{s_0}) - (s_1 + s_0) \\ &= \widehat{s_0} - s_0 = \epsilon \star f_0 \bmod q. \end{aligned}$$

The attacker can obtain the secret key f_0 as

$$f_0 = \epsilon^{-1} \star \Delta s \bmod q$$

by exhaustively trying all possible $(q^t - 1)\binom{N}{t} = \mathcal{O}((qN)^t)$ values for $\epsilon(x)$ with the pre-specified small number of non zero coefficients. Calculating the multiplicative inverse of ϵ in R_q can be done efficiently using the Newton iteration method [35]. Then, the attacker calculates

$$g_0 = f_0 \star h \bmod q,$$

Table 1 Experimental results for NTRUSign with $(N, q, df, dg, \mathcal{B}, \text{“type”}, \mathcal{N}) = (251, 128, 73, 71, 0, \text{standard}, 310)$

t	ϵ is non-invertible	ϵ is invertible		Probability of successful fault injection
		Fail	Pass	
1	4901	3153	1946	0.1946
2	4917	4563	520	0.0520
3	5093	4789	118	0.0118

The attacker can verify the correctness of the obtained solution by running the verification algorithm on signatures produced by these recovered keys.

The success of the above attack requires that ϵ is invertible in R_q. For $q = p^l$ is a power of a prime p and N is a prime number, let $n \geq 1$ be the smallest integer such that $p^n \equiv 1 \bmod N$. The probability that $\epsilon_q(x) \in R_q$ is invertible is given by [34]

$$\left(1 - \frac{1}{p}\right)\left(1 - \frac{1}{p^n}\right)^{(N-1)/n}$$

which is approximately equal to $(1 - \frac{1}{p})$ for the typical choice of the parameters of the NTRUSign algorithm. For example, for $N = 251$ and $q = 128$ [2], we have $p = 2$ and $n = 50$. Consequently, the probability that an element in R_q is invertible $\approx \frac{1}{2}$.

Our analysis above ignores the fact that the signature procedure might be implemented in such a way that the attacker may not have access to the faulty signature if it does not pass the norm-bound checking (step 6 in the signing procedure in Algorithm 1). While the NTRUsign designers have previously suggested that, depending on the application, this step might be skipped for NTRUSign with no perturbation in order to improve the efficiency of the signing process (see Section 6.4 in [1]), this is not the case for NTRUSign with $\mathcal{B} > 0$.

Tables 1 and 2 show the percentage of cases in which the faulted signatures pass this check for 10 random messages and 1,000 fault injections for each message for the parameter set $(N, q, df, dg, \mathcal{B}, \text{“type”}, \mathcal{N}) = (251, 128, 73, 71, \mathcal{B}, \text{standard}, 310)$ [2] for $\mathcal{B} = 0$ and $\mathcal{B} = 1$, respectively. In the above set of parameters, N denotes the dimension of the polynomial ring used, i.e., polynomials are up to degree $N - 1$, $\mathcal{N}$ denotes the norm-bound, df, dg denote the Hamming weight of the binary polynomials f and g used in the key generation step of the algorithm, $\mathcal{B} = 0$ corresponds to the system with no perturbation and $\mathcal{B} = 1$ corresponds to the system with perturbation (see Algorithm 1 for further clarification of these parameters.) The experimental results were obtained by simulating the process of fault injection in our MAPLE software implementation of NTRUSign.

From Table 1, if the number of faulted coefficients in A is 1, then about $1/0.195 \approx 5$ fault injections are required before the attacker is able to access the faculty signature which corresponds to an invertible error polynomial $\epsilon \in R_q$. It is also clear that the required number of fault injections increases with t. If, for practical

Table 2 Experimental results for NTRUSign with $(N, q, df, dg, \mathcal{B}, \text{“type”}, \mathcal{N}) = (251, 128, 73, 71, 1, \text{standard}, 310)$

t	ϵ is non-invertible	ϵ is invertible		Probability of successful fault injection
		Fail	Pass	
1	4948	4051	1001	0.1001
2	4990	4857	153	0.0153
3	5016	4962	22	0.0022

Table 3 Experimental results for NTRUSign with $(N, q, df, dg, \mathcal{B}, \text{“type”}, \mathcal{N}) = (251, 128, 73, 71, 0, \text{transpose}, 310)$

t	ϵ is non-invertible	ϵ is invertible		Probability of successful fault injection
		Fail	Pass	
1	4944	4437	619	0.0619
2	5064	4893	43	0.0043
3	4089	5008	3	0.0003

reasons, inducing such a large number of transient fault injections is not possible, then a second order fault attack [27] can be utilized in order to reduce the number of required fault injections. In this case, the attacker induces a second fault in step 6 of Algorithm 1 to skip this norm-bound check.

Similar analysis follows for type = “transpose”, i.e., when the signing process is performed using the NTRU transpose lattice. However, in this case, g_0 cannot be recovered from the knowledge of the public key and f_0. Thus the attacker has to repeat the attack by injecting another fault in the computation of the polynomial B (step 6 in Fig. 1). Tables 3 and 4 show the corresponding experimental results for NTRUSign with the transpose lattice.

The above set of experiments corresponds to the security parameters defined in the Efficient Embedded Security Standards (EESS) standard [4] with a claimed security level equivalent to RSA-1024 which is roughly equivalent in strength to 80-bit symmetric keys [4]. In order to test the effect of changing the algorithm claimed theoretical security level on the performance of our attack when the attacker cannot bypass the signature verification step, we performed another set of experiments on NTRUSign with the new set of security parameters defined in [33, 34]. For this version of NTRUSign, the centered norm computation in step 3 of the verification procedure in Algorithm 1 is defined as $\|s, s \star h - m_0\|^2 = \|s\|^2 + \beta^2 \|s \star h - m_0\|^2$ where, as shown in Table 5, different values of β are defined for each set of the security parameters (see also Section 2.3 in [34]) It should be noted that these parameters are only applicable to the NTRUsign with the transpose NTRU lattice. Furthermore, in this version of NTRUSign, the polynomials f and g have trinary coefficients $\in \{+1, -1, 0\}$ with $(d+1)$ $+1$ and (d) -1 while in the EESS version of NTRUSign, f and g are binary polynomials with Hamming weight df and dg, respectively. The first column in Table 5 refers to the claimed security level, k, in bits. One interesting observation, that follows from the experimental results in Table 5, is that increasing the security parameter, which basically reflects the resistance against the best known lattice based attacks, does not necessarily improve the resistance against our attack. It should also be noted that if the attacker is able to skip the norm-bound checking step, then the probability of successful fault insertion $\approx 1 - \frac{1}{p}$, does not vary with the choice of the security parameters since $p = 2$ for all of them. On the other hand, as depicted in the table, in the case where the attacker cannot skip the norm-bound checking step, the precision of the fault injection process, i.e.,

Table 4 Experimental results for NTRUSign with $(N, q, df, dg, \mathcal{B}, \text{“type”}, \mathcal{N}) = (251, 128, 73, 71, 1, \text{transpose}, 310)$

t	ϵ is non-invertible	ϵ is invertible		Probability of successful fault injection
		Fail	Pass	
1	4948	4775	277	0.0277
2	5026	4965	9	0.0009
3	5049	5051	0	0.0000

Table 5 Experimental results for NTRUSign with the set of parameters in [33, 34]

Parameters						Probability of successful fault injection		
k	N	d	q	β	$\mathcal{N}$	$t=1$	$t=2$	$t=3$
80	157	29	256	0.38407	150.02	0.0303	0.0012	0.0000
112	197	28	256	0.51492	206.91	0.0241	0.0011	0.0001
128	223	32	256	0.65515	277.52	0.0266	0.0012	0.0000
160	263	45	512	0.31583	276.53	0.0188	0.0007	0.0000
192	313	50	512	0.40600	384.41	0.0221	0.0008	0.0000
256	349	75	512	0.18543	368.62	0.0350	0.0031	0.0000

the ability of the attacker to bound the number of faulted polynomial coefficients, t, has a large influence on the success probability of the attack and also keeps the attack required computation steps, $O((qN)^t)$, in the practical range.

4 Countermeasures against fault analysis of NTRUSign

To secure cryptographic devices against fault analysis, proper countermeasures have to be applied. Generally, these countermeasures try to detect any temporal or permanent faults which happen in the cryptosystem, and then, immediately, disable the device output or rest all the output bits to 0s. As a result, the attacker is prevented from observing the output of the faulty cryptographic computations and hence the vulnerability of the cryptosystem to these attacks can be alleviated. Several approaches of fault detection techniques have been investigated. These techniques include error detecting codes and redundancy-based techniques [37].

Unlike the case of symmetric key ciphers where parity based error detection codes are somewhat straightforward to derive for the intermediate computation steps of the cipher, this does not seem to be the case for NTRUsign because of the nonlinearity of the operations involved, especially the rounding operation, $\lfloor \cdot \rceil$, in steps 2 and 6 of the algorithm in Fig. 1. On the other hand, the algebraic properties of the polynomials convolutional product defined by (1), allow us develop a redundancy-based fault detection technique which detects both transient and permanent faults. We have also developed another, independent, technique to defend against second order fault analysis where the attacker is able to introduce another fault to skip the fault detection check.

4.1 Recomputing with cyclically shifted messages

Let $a^{(r)}(x) \in R$ denote an r-cyclically shifted version of $a(x) = \sum_i a_i x^i$, $0 < r \leq N-1$. The coefficients of $a^{(r)}(x)$ are obtained by rotating the coefficients of $a(x)$ by r positions. Thus we have $a^{(r)}(x) = \sum_{i=0}^{N-1} a_{i+r \,(\mathrm{mod}\ N)} x^i$. Let $c(x) = a(x) \star b(x)$. Then we have $a^{(r)}(x) \star b(x) = c^{(r)}(x)$ where $c_i^{(r)} = c_{i+r \,(\mathrm{mod}\ N)}$.

Consequently, the coefficients of $\lfloor \frac{a^{(r)}(x) \star b(x)}{q} \rceil$ correspond to r-cyclically shifted version of the coefficients of $\lfloor \frac{a(x) \star b(x)}{q} \rceil$. This fact can be utilized to introduce either spatial redundancy or temporal redundancy where the redundant computation is

performed using a rotated version of m and the two signatures are compared as shown in Fig. 2. If the result of the two operations do not match, up to the chosen rotation r, then the output of the device is disabled in order to prevent the attacker from accessing any information from the faulty signature. Note that this technique has an advantage over straightforward application of temporal redundancy, i.e., evaluating the signature on the same message for two times and comparing the result, since the latter approach does not protect against permanent faults. To speed up computation in the case of utilizing the temporal redundancy option, the added redundancy can be utilized at the level of smaller building blocks, i.e., during the computation of A and B in step 6 of Fig. 1. However, applying this approach at the algorithm level has the advantage of being able to detect all faults, not necessarily the ones resulting from injecting errors at this particular step.

4.2 Defending against second order fault analysis attacks

In this section, we introduce another countermeasure which avoids decision test (i.e., the error checking that is performed by comparing the signature with the one resulting from the redundant module such as the one shown in Fig. 2) and consequently protects against sophisticated attackers who are able to induce higher order faults that skip fault detection operations. Our approach is inspired by the fault infective computation technique proposed by Yen et al. [28] to defend against fault attacks on the RSA digital signature scheme.

The following two lemmas will be used to prove the correctness of our approach:

Lemma 1 *Let* $w = \frac{z}{q} - \lfloor \frac{z}{q} \rceil$, *where* z *and* q *are two integers. If an error occurs during the computation of* $\lfloor \frac{z}{q} \rceil$, *then we have* $\lfloor w \rceil \neq 0$.

Proof Let w and w' denote the results of correct and incorrect computations, respectively. From to the definition of the round function $\lfloor \cdot \rceil$, $w \in [-\frac{1}{2}, \frac{1}{2})$. Since erroneous $\lfloor \frac{z}{q} \rceil$ differs from correct $\lfloor \frac{z}{q} \rceil$ by an integer, the difference between w and w'

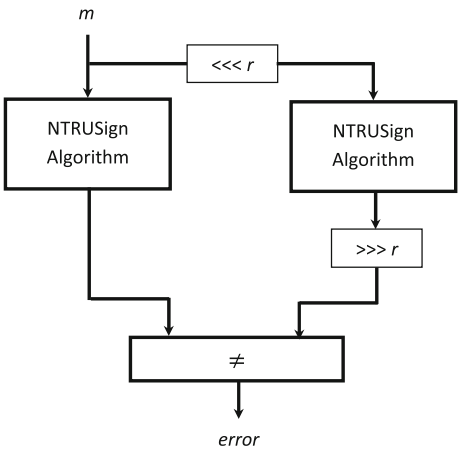

Fig. 2 Fault detection by recomputing with cyclically shifted messages

is given by $|w - w'| \geq 1$. Thus, $w' \notin [-\frac{1}{2}, \frac{1}{2})$ and consequently $\lfloor w' \rceil \neq 0$, which proves the lemma. □

Lemma 2 *If z_1, z_2 are two nonnegative integers, then*

$$\left\lceil \frac{z_1 + z_2}{(z_1 + z_2 + 1)} \right\rceil = \begin{cases} 0 & \textit{if } z_1 = z_2 = 0 \\ 1 & \textit{otherwise}. \end{cases}$$

Proof Let $z = z_1 + z_2$. It is clear that $\frac{z}{z+1} = 0$ for $z = 0$ and $\frac{1}{2} \leq \frac{z}{z+1} < 1$ for $z \geq 1$. It follows that $\lceil \frac{z}{z+1} \rceil = 0$ if $z = 0$ and $\lceil \frac{z}{z+1} \rceil = 1$ otherwise. The lemma follows by noting that $z = 0$ if and only if $z_1 = z_2 = 0$. □

To defend against higher order fault analysis of NTRUSign, step 5 in the signing procedure of Algorithm 1 is replaced by the following steps:

a. Calculate $A = \lfloor \frac{-f_0' \star m}{q} \rceil$, $B = \lfloor \frac{f_0 \star m}{q} \rceil$,
b. Calculate $AA = \lfloor \frac{-f_0' \star m}{q} - A \rceil$, $BB = \lfloor \frac{f_0 \star m}{q} - B \rceil$,
c. Set $s_0 = A \star f_0 + B \star f_0' \bmod q$,
d. Set $z_1 = \sum_{i=0}^{N-1} |AA_i|$, $z_2 = \sum_{i=0}^{N-1} |BB_i|$, where $|\cdot|$ denotes the absolute value of the enclosed coefficient.
e. Set $s = q^{\lceil \frac{z_1+z_2}{z_1+z_2+1} \rceil}(s + s_0) \bmod q$, where $\lceil \cdot \rceil$ denotes the ceiling operation.

From Lemma 1, the coefficients of the polynomials AA and BB will all be equal to zeros if and only if no errors occur during the computation of this step. Consequently, and by utilizing Lemma 2, in case of no errors, step (e) above evaluates

$$s = q^{\lceil \frac{z_1+z_2}{z_1+z_2+1} \rceil}(s + s_0) \bmod q = q^0 \times (s + s_0) \bmod q = (s + s_0) \bmod q$$

which corresponds to the correct output of Algorithm 1.

On the other hand, if an error occurs, step (e) above results in

$$s = q^{\lceil \frac{z_1+z_2}{z_1+z_2+1} \rceil}(s + s_0) \bmod q = q \times (s + s_0) \bmod q = 0,$$

which prevents the attacker from recovering any useful information about the secret key. Note that we raise q to the power of $\lceil \frac{z_1+z_2}{z_1+z_2+1} \rceil \in \{0, 1\}$ instead of raising it to the power of $z_1 + z_2$ in order to avoid the unnecessarily computational complexity of performing the exponentiation operation with a large exponent.

5 Conclusions

We presented a fault analysis attack on the NTRU public key digital signature algorithm (NTRUSign). One interesting observation is that the perturbation process, which was introduced to improve the security of NTRUSign towards previous attacks, does not improve the resistance of NTRUSign against this kind of fault attacks if the norm-bound checking step can be skipped by the attacker. Another observation is that using NTRUSign with a larger-security parameter, while improving the resistance against lattice based attacks, does not necessarily improve the resistance against fault analysis attacks especially in the cases where the attacker

is able to precisely corrupt small number of coefficients and is able to skip the norm-bound checking step. Thus, when NTRUSign is deployed in environments that are susceptible to this class of fault analysis attacks, implementing fault analysis countermeasures, such as the two countermeasures developed in this work, is necessary to protect the security of the system even if the largest-security parameters are utilized.

Acknowledgements The authors would like to thank the anonymous reviewers for their valuable comments and suggestions that helped improve the quality of the paper. This work is supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

1. Hoffstein, J., Graham, N., Pipher, J., Silverman, J., Whyte, W.: NTRUSign: Digital signatures using the NTRU lattice. Draft 2, NTRU Cryptosystem Inc. (2002). Available at: www.sisecure.com/cryptolab/pdf/NTRUSign-preV2.ps
2. Hoffstein, J., Graham, N., Pipher, J., Silverman, J., Whyte, W.: NTRUSign: digital signatures using the NTRU lattice. In: Proc. of CT-RSA'03, LNCS 2612, pp. 122–140. Springer (2003)
3. Hoffstein, J., Pipher, J., Silverman, J.: An Introduction to Mathematical Cryptography. Undergraduate Texts in Mathematics. Springer (2008)
4. Consortium for Efficient Embedded Security. Efficient Embedded Security Standard (EESS)#1: Implementation Aspects of NTRUEncrypt and NTRUSign (2003). Available at http://grouper.ieee.org/groups/1363/lattPK/submissions/EESS1v2.pdf
5. Hoffstein, J., Pipher, J., Silverman, J.: NSS: an NTRU lattice-based signature scheme. In: Proc. of EUROCRYPT'01, LNCS 2045, pp. 211–228. Springer (2001)
6. Gentry, C., Jonsson, J., Stern, J., Szydlo, M.: Cryptanalysis of the NTRU signature scheme (NSS) from Eurocrypt 2001. In: Proc. of ASIACRYPT'01, LNCS 2248, pp. 1–20. Springer (2001)
7. Gentry, C., Szydlo, M.: Cryptanalysis of the revised NTRU signature scheme. In: Proc. of EUROCRYPT'02, LNCS 2332, pp. 299–320. Springer (2002)
8. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Proc. of CRYPTO'97, LNCS 1294, pp. 112–131. Springer (1997)
9. Min, S., Yamamoto, G., Kim, K.: Weak property of malleability in NTRUSign. In: Proc. ACISP'04, LNCS 3108, pp. 379–390. Springer (2004)
10. Szydlo, M.: Hypercubic lattice reduction and analysis of GGH and NTRU signatures. In: Proc. of EUROCRYPT'03, LNCS 2656, pp. 433–448. Springer (2003)
11. Nguyen, P., Regev, O.: Learning a parallelepiped: cryptanalysis of GGH and NTRU signatures. In: Proc. of EUROCRYPT'06, LNCS 4004, pp. 215–233. Springer (2006)
12. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. In: Proc. of EUROCRYPT'97, LNCS 1233, pp. 37–51. Springer, Heidelberg (1997)
13. Biernat, J., Nikodem, M.: Fault cryptanalysis of ElGamal signature scheme. In: Proc. of EUROCAST'05, LNCS 3643, pp. 327–336. Springer (2005)
14. Giraud, C., Knudsen, E., Tunstall, M.: Improved fault analysis of signature schemes. In: Proc. of CARDIS'10, LNCS 6035, pp. 164–181. Springer (2010)
15. Biehl, I., Meyer, B., Muller, V.: Differential fault analysis on elliptic curve cryptosystems. In: Proc. of CRYPTO'00, LNCS 1880, pp. 131–146. Springer (2000)
16. Seifert, J.: On authenticated computing and RSA-based authentication. In: Proc. of ACM CCS'05, pp. 122–127. ACM Press (2005)
17. Muir, J.: Seifert's RSA fault attack: simplified analysis and generalizations. In: Proc. of ICICS'06, LNCS 4307, pp. 420–434. Springer (2006)
18. Brier, E., Chevallier-Mames, B., Ciet, M., Clavier, C.: Why one should also secure RSA public key elements. In: Proc. of CHES'06, LNCS 4249, pp. 324–338. Springer (2006)
19. Berzati, A., Canovas, C., Goubin, L.: Perturbating RSA public keys: an improved attack. In: Proc. of CHES'08, LNCS 5141 , pp. 380–395. Springer (2008)
20. Berzati, A., Canovas, C., Doumas, J., Goubin, L.: Fault attacks on RSA public keys: left-to-right implementations are also vulnerable. In: Proc. of CT-RSA'09, LNCS 5473, pp. 414–428. Springer (2009)
21. Biham, E., Shamir, A.: Differential fault analysis of secret key cryptosystems. In: Proc. of CRYPTO'97, LNCS 1294, pp. 513–525. Springer (1997)

22. Dusart, P., Letourneux, G., Vivolo, O.: Differential fault analysis on AES. In: Proc. of ACNS'03, LNCS 2846, pp. 293–306. Springer (2003)
23. Hoch, J., Shamir, A.: Fault analysis of stream ciphers. In: Proc. of CHES'04, LNCS 3156, pp. 240–253. Springer (2004)
24. Kamal, A., Youssef, A.: Fault analysis of NTRUEncrypt. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E94-A**(4), 1156–1158 (2011)
25. Blömer, J., Otto, M.: Wagner's attack on a secure CRT-RSA algorithm reconsidered. In: Proc. of FDTC'06, LNCS 4236, pp. 13–23. Springer (2006)
26. Shamir, A.: Method and apparatus for protecting public key schemes from timing and fault attacks. United States Patent #5991415, November 23, 1999. Also presented at the rump session of EUROCRYPT'97
27. Kim, C., Quisquater, J.: Fault attacks for CRT based RSA: new attacks, new results, and new countermeasures. In: Proc. of WISTP'07, LNCS 4462, pp. 215–228. Springer (2007)
28. Yen, S., Kim, S., Lim, S., Moon, S.: RSA speedup with Chinese Remainder Theorem immune against hardware fault cryptanalysis. IEEE Trans. Comput. **52**(4), 461–472 (2003)
29. Blömer, J., Otto, M., Seifert, J.: A new CRT-RSA algorithm secure against Bellcore attacks. In: Proc. of CCS'03, pp. 311–320 (2003)
30. Ciet, M., Joye, M.: Practical fault countermeasures for Chinese remaindering based RSA. In: Proc. of FDTC'05, pp. 124–131 (2005)
31. Driessen, B., Poschmann, A., Paar, C.: Comparison of innovative signature algorithms for WSNs. In: Proc. of WiSec'08, pp. 30–35. ACM Press (2008)
32. Skorobogatov, S., Anderson, R.: Optical fault induction attacks. In: Proc. of CHES'03, LNCS 2523, pp. 2–12. Springer (2003)
33. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Whyte, W.: Practical lattice-based cryptography: NTRUEncrypt and NTRUSign. In: The LLL Algorithm, pp. 1–42. Springer, Berlin (2010)
34. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J., Whyte, W.: Performance improvements and a baseline parameter generation algorithm for NTRUSign. In: Workshop on Mathematical Problems and Techniques in Cryptology, Barcelona, Spain (2005)
35. Silverman, J.: Almost inverses and fast NTRU key creation. NTRU Report 014, NTRU Cryptosystem Inc. (1999). Available at: http://securityinnovation.com/cryptolab/pdf/NTRUTech014.pdf
36. Silverman, J.: Invertibility in truncated polynomial rings. NTRU Report 009, NTRU cryptosystem Inc. (1998). Available at: http://securityinnovation.com/cryptolab/pdf/NTRUTech009.pdf
37. Koren, I., Mani Krishna, C.: Fault-Tolerant Systems. Elsevier/Morgan Kaufmann (2007)