A New Signature Using Lattices With a Security Reduction in The Random Oracle Model

Abstract. In this paper we propose a signature scheme over truncated polynomial rings using lattices. It is similar to NTRUSign and NSS but contrary to those schemes, this system is proved to be equivalent to his internal one-way function via a tight security reduction in the random oracle model. As NEAP or Lyubashevsky lattice based-signatures, this scheme is a quantum resistant cryptosystem.

Keywords: Lyubashevsky lattice based-signatures, Post-quantum cryptography, NEAP, NTRU, NTRUencrypt, NTRUsign, NSS, PSS-R, Provable security, Lattice attack, Short vector problem, Transcript attack, Meet-in-the-Middle attack.

1 Introduction

Until now, there exists a few efficient signatures based on lattices and rigorously proven secure (see [2], [29], [4], [19], [19]). According the role of digital signatures for information security, it will always be interesting to find a new signature scheme with a tight security reduction or to improve the security of a known one.

A proof of security of a signature scheme generally proceeds by demonstrating that if a polynomial-time adversary $\mathcal{A}$ is able to break the signature scheme, it can be used by a reduction algorithm $\mathcal{B}$ to invert in polynomial time some related one-way-function.

Given an attacker $\mathcal{A}$ which is able to break the signature in time τ_A with success probability at least ε_A, for the reduction proof, $\mathcal{B}$ must simulate the environment of $\mathcal{A}$ and solves the problem (invert the one way function) with time $\tau_B \geq \tau_A$ and success probability $\varepsilon_B \leq \varepsilon_A$.

For tightness of the reduction it is required to have $\varepsilon_B \approx \varepsilon_A$ and $\tau_B \approx \tau_A + polynom(k)$ where k is a security parameter.

The main advantage of lattice-based signatures is that these schemes are in general quantum resistant signatures algorithms. But, there exists only a few such signatures.

In the standard oracle model, there exists some signatures using lattices with a very strong security proofs such as those of Boyen [2], Ruckert [29], and Cash et al. [4]. But, as pointed out by some researchers, these previous signatures are somewhat unpractical, and of mainly theoretical interest.

In the random oracle model, there exists practical lattice-based signatures with security proofs (see: [19], [19]). But the design of these schemes are quite different from those of NTRU and NTRUsign.

In this work, we propose a signature scheme provably secure in the random oracle model and based on NTRU quotient rings. It is similar to NTRUsign [13] and NSS [12] but contrary to those schemes, this system is proved to be equivalent to his internal one-way function via a tight security reduction. The security proof is based on a technic which use ideas from PSS-R [6] of Coron and et *al.*

This paper is organized as follows:
- Section 1: Mathematical model
- Section 2: Provable security

2 Mathematical model

The signature presented here use quotient rings and lattices as NTRU and NTRUSign (see "NTRU reports" [23]).

- Let p, r, q and N be four prime integers and c a small integer. We consider the following rings: $\mathcal{R} = \frac{\mathbb{Z}[x]}{(x^N-1)}$, $\mathcal{R}_q = \frac{\frac{\mathbb{Z}}{q\mathbb{Z}}[x]}{(x^N-1)}$, $\mathcal{R}_r = \frac{\frac{\mathbb{Z}}{r\mathbb{Z}}[x]}{(x^N-1)}$,, $\mathcal{R}_c = \frac{\frac{\mathbb{Z}}{c\mathbb{Z}}[x]}{(x^N-1)}$, and $\mathcal{R}_p = \frac{\frac{\mathbb{Z}}{p\mathbb{Z}}[x]}{(x^N-1)}$

- A polynomial $f(X) = f_0 + f_1X + \ldots + f_{N-1}X^{N-1} \in \mathcal{R}$ is identified with its vector of coefficients $f = [f_0; f_1; \ldots; f_{N-1}]$. The Euclidean norm $||f||$ of a polynomial or vector is defined by: $||f||^2 = \sum_{i=0}^{N-1} f_i^2$. We have $||(f,g)||^2 = ||f||^2 + ||g||^2$.

- The width of f is $\text{Width}(f) = Max(f_0; f_1; \ldots; f_{N-1}) - Min(f_0; f_1; \ldots; f_{N-1})$, the support of f is $\text{Supp}(f) = \{f_i / f_i \neq 0\}$ and it's cardinality is denoted by $d_f = \#\text{Supp}(f)$.

- We consider throughout this paper the following three sets $\mathcal{L}_{\text{key}}, \mathcal{L}_{\text{rand}}, \mathcal{L}_{\text{text}}$ of polynomials in the ring $\mathcal{R}$, in which we are going to choose public keys, random and messages.

- Define the sets:

$\mathbb{L}_1(\delta) = \{f \in R/f$ has δ coeff equal to $-2, -1, 2$; δ+1 coeff equal to 1, and $N-4\delta-1$ coeff equal to $0\}$,

$\mathbb{L}_2(\delta) = \{f \in R/f$ has δ coeff equal to -1, $\delta+1$ coeff equal to 1, and $N-2\delta-1$ coeff equal to $0\}$,

and $\mathbb{B}(\eta) = \{f \in R/f$ has η+1 coeff equal to 1 and $N-\eta-1$ coeff equal to $0\}$.

- In general, we assume that $\mathcal{L}_{\text{key}} \subset \mathbb{L}_1(\delta)$ or $\mathbb{L}_2(\delta)$ and $\mathcal{L}_{\text{rand}}, \mathcal{L}_{\text{text}} \subset \mathbb{B}(\eta)$.

- If $f \in \mathbb{L}_1(\delta)$, $\big($respectively : $f \in \mathbb{L}_2(\delta)$, $f \in \mathbb{B}(\eta)\big)$ then we have $||f|| = \sqrt{10\delta+1}$, $\big($respectively :$||f|| = \sqrt{2\delta+1}$, $||f|| = \sqrt{\eta}\big)$.

2.1 Signature algorithm

In this subsection, we describe the key generation algorithm, the signature mechanism and the verification process. We use two entities for signature and verification: Ngary and Nague. Throughout this paper, we use "NSULROM" to abbreviate the long title of this article.

Key Generation

To create a public/private key, Ngary should do the following.

1. Chooses six integers p, c, r, q , d_s and N with $r < q$, $2 < d_s < N$ and p, r, q and N primes;
2. Chooses at random three polynomials $f, g \in \mathcal{L}_{\text{key}}$, and $u \in \mathcal{L}_{\text{rand}}$ invertible mod q, that is $f_q * f \equiv 1 \mod q$, $g_q * g \equiv 1 \mod q$ and $u_q * u \equiv 1 \mod q$;
3. Chooses at random a polynomial $\bar{t} \in \mathcal{L}_{\text{rand}}$ and defines $t = 1 + c\bar{t}$
4. Computes $h_0 = t * u_q * f_q \mod q$, and $h_1 = t * u_q * g_q \mod q$; the polynomial t may be discarded.
5. $\big((h_0, h_1), (p, c, r, q, d_s)\big)$ is the public key for verification and $\big((f, g, u), (p, c, r, q, d_s)\big)$ is the private key for signature.

Signature

To sign a message $m \in \mathcal{L}_{\text{text}}$ or $m = (m_1, m_2) \in \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$ for Nague , Ngary should do the following;

1. obtain his private key $\big((f, g, u), (p, c, r, q, d_s)\big)$;
2. selecting a message m from the set of signing texts $\mathcal{L}_{\text{text}}$;
3. chooses randomly three polynomials $\phi, \varphi, \psi \in \mathcal{L}_{\text{rand}}$; and compute: $\Gamma_0 = f * [u * (\widetilde{m} + \phi) + \psi] \in \mathcal{R}$ and $\Gamma_1 = g * [u * (\phi + c.\varphi) + \psi] \in \mathcal{R}$ with $\widetilde{m} = m$ if $m \in \mathcal{L}_{\text{text}}$ or $\widetilde{m} = m_1 + pm_2$ if $m = (m_1, m_2) \in \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$
4. if $\Gamma_0 \mod r \neq \Gamma_0$ or $\Gamma_1 \mod r \neq \Gamma_1$ return to (3)
5. if $\min(d_{\Gamma_0}, d_{\Gamma_1}) < d_s$ return to (3)
6. (Γ_0, Γ_1, m) is the signature which Ngary sends to Nague.

Verification

In order to verify the signature (Γ_0, Γ_1, m), Nague should do the following;

1. obtains Ngary's public key $\big((h_0, h_1), (p, c, r, q, d_s)\big)$;
2. if $\Gamma_0 \mod r \neq \Gamma_0$ or $\Gamma_1 \mod r \neq \Gamma_1$ reject the signature;
3. if $\min(d_{\Gamma_0}, d_{\Gamma_1}) < d_s$ reject the signature,
4. computes: $A_s = h_0 * \Gamma_0 - h_1 * \Gamma_1 \mod q$
5. chooses the coefficients of A_s to lie in an interval of the form $-\frac{q-1}{2}$ to $\frac{q-1}{2}$
6. computes $B_s = A_s \mod c$
7. Accepts the signature if and only if $B_s = m$ for $m \in \mathcal{L}_{\text{text},p}$ or $B_s = m_1 + pm_2$ for $m = (m_1, m_2) \in \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$

Why verification works?

With the public key $h_0 = t * u_q * f_q \mod q$, and $h_1 = t * u_q * g_q \mod q$, we have

$A_s = h_0 * \Gamma_0 - h_1 * \Gamma_1 = h_0 * f * [u * (\widetilde{m} + \phi) + \psi] - h_1 * g * [u * (\phi + c.\varphi) + \psi] \mod q$

hence $A_s = t * [(\widetilde{m} + \phi) + u_q\psi] - t * [(\phi + c.\varphi) + u_q\psi] \mod q$

$A_s = t * (\widetilde{m} + c.\varphi) \mod q$

Now, let us choose the coefficients of A_s to lie in the interval from $-\frac{q-1}{2}$ to $\frac{q-1}{2}$, then this is an equality over $\mathbb{Z}$, rather than just over $\frac{\mathbb{Z}}{q\mathbb{Z}}$:

Now, let us compute $B_s = A_s \mod c \equiv t * (\widetilde{m} + c.\varphi) \mod c = \widetilde{m} \mod c$.

At the end, accept the signature if and only if $B_s = m$ for $m \in \mathcal{L}_{\text{text}}$ or $B_s = m_1 + pm_2 +$ for $m = (m_1, m_2) \in \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$

2.2 One-way Trapdoor function

Consider the following maps:

Signature map: Let $\mathcal{S}_{r,d_s} = \{\Gamma \in \mathcal{R}_r / d_\Gamma \geq d_s\}$.

the signature function is the following; $\mathbb{L} \times L^3_{\text{rand}} \xrightarrow{\mathcal{S}} \mathcal{S}_{r,d_s} \times \mathcal{S}_{r,d_s}$: with $\mathcal{S}\big(m;(\phi,\varphi,\psi)\big) = (\Gamma_0,\Gamma_1)$ where

$\Gamma_0 = f*[u*(\widetilde{m}+\phi)+\psi] \in \mathcal{S}_{r,d_s}$ and $\Gamma_1 = g*[u*(\phi+c.\varphi)+\psi] \in \mathcal{S}_{r,d_s}$ with $\widetilde{m} = m$ if $m \in \mathbb{L} = \mathcal{L}_{\text{text}}$ or $\widetilde{m} = m_1 + pm_2 +$ if $m = (m_1,m_2) \in \mathbb{L} = \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$

One-way function: verification map

$\mathcal{S}_{r,d_s} \times \mathcal{S}_{r,d_s} : \xrightarrow{\mathcal{V}} \mathcal{R}_q : (\Gamma_0,\Gamma_1) :\mapsto \mathcal{V}(\Gamma_0,\Gamma_1) = h_0 * \Gamma_0 - h_1 * \Gamma_1 \mod q$

where h_0, h_1 are given by the output of key generation.

Note that we have $y = h_0*\Gamma_0 - h_1*\Gamma_1 \mod q \Leftrightarrow y*h_1^{-1} = h_1^{-1}*h_0*\Gamma_0 - \Gamma_1 \mod q \Leftrightarrow y*h_0^{-1} = \Gamma_1 - h_0^{-1}*h_1*\Gamma_1 \mod q$ with $h_1^{-1}*h_0 = g*f_q \mod q$. This one-way function is similar to those of NTRUencrypt ([14] and [15]) but is slightly different from NTRUencrypt one-way function because of the form of the domain of the function.

2.3 Attacks against the cryptosystem

Forgery attack for the signature :

Let $(\Gamma_0,\Gamma_1) \in \mathcal{S}_{r,d_s} \times \mathcal{S}_{r,d_s}$, put $A = \mathcal{V}(\Gamma_0,\Gamma_1)$ and define $m = A \mod c$, then (Γ_0,Γ_1,m) is a valid signature.

NB: To protect against this attack and all kind of chosen cipher texts attacks, we are going in the next section to design a new version with a security proof by adapting the PSS-R model [6].

Transcript Attack for the signature :

We can circumvent transcript attack by using hash functions (see [10], [13] and [7] and [8]) and by designing a new model with a tight security reduction in the next section. With this new model, the recent improvements on attacks based on leakage information on the private key ([25] and [26]), do not work.

Attacks against the public key .

Recall that $h_0 = t*u_q*f_q \mod q$ and $h_1 = t*u_q**g_q \mod q$ (with $t = 1+c\bar{t}$) is the public key.

We have the three equations;

(1) $(u*f)*h_0 \mod q = t = (u*g)*h_1 \mod q$,

(2) $f*(h_1^{-1}*h_0) = g \mod q \Leftrightarrow f*\overline{h} = g \mod q$ with $\overline{h} = h_1^{-1}*h_0 \mod q$

(3) $u*(f*h_0) \mod q = t = u*(g*h_1) \mod q$ (if f and g are known)

Lattice attack against the public key:

For equation **(2)**:

Let θ be a small value. Let $A = (A_0,\dots A_{N-1})$ and $B = (B_0,\dots B_{N-1})$ be two polynomials. Define a lattice $\mathfrak{L}_{key} = \{(\theta A, B)/A*(h_1^{-1}*h_0) = t = 1+c\bar{t} \mod q$, thus (f,g) is an element of $\mathfrak{L}_{key}$. This lattice has dimension $2N$.

As in classical NTRU, the optimal value of θ is $\theta = \frac{||f||}{||g||}$ and this case the value:

$$c_{\overline{h}} = \frac{|\text{target vector}|}{|\text{expected shortest vector}|} = \sqrt{\frac{2\pi e||f||.||g||}{Nq}}$$

measure how far the associated lattice departs from a random lattice. On must choose the different polynomials such that $c_{\overline{h}} \sim 1$

Also, as in classical NTRU, it is difficult to make $c_{\overline{h}}$ close to 1, if we use $3-ary$ or $5-ary$ polynomials. Nevertheless, solving the lattice problem is difficult if N is large ($N > 440$ for Type = **A** and $N > 300$ for Type = **B**) and d_f and d_g are not too small.

For equation **(1)** It is easy to see that this equation is more difficult to solve, hence it will suffice to choose f and g such that lattice attack fail for equation **(2)** and choose u and t as in the algorithm at the end of the next section.

remark For NTRUencrypt and NTRUsign, any closed vector (f', g') of (f, g) allows to decrypt or to sign but is not the case for this scheme because of equation **(3)**: if $f' * \overline{h} = g' \mod q$ and $u' * (f' * h_0) \mod q = t'$ then (u', t') must verify also $u' * (g' * h_1) \mod q = t'$, which is hard to obtain. It is the reason why we claim (without proof) that the security is based on Short Vector Problem (SVP) rather than Closed Vector Problem (CVP).

Meet-in-the-Middle attack against the public key.

1) With the public key $\overline{h} = h_1^{-1} * h_0$, we have $f * \overline{h} = g \mod q$, and an attacker can splits f in half, say $f = f_1||f_2$ and write $f = f_1 + f_2$ (by adding $N/2$ zeros to f_i to have a vector of size N), and compute $f_1 * \overline{h} = g - f_2 * \overline{h} \mod q$ thus

$(f_1 * \overline{h})_i = \{-\frac{a-1}{2} + 1, \ldots - 1, 1 \ldots fraca - 12\} - (f_2 * \overline{h})_i(\mod q)$, $\forall\ 0 < i \leq N - 1$, with $a = 3$ or 5. Using technic from [16], we see that for $a = 5$, the attacker must perform $\text{Comb}_{(f,\overline{h})}$ operations with:

$$\text{Comb}_{(f,\overline{h})} \simeq \frac{\left[\binom{N/2}{\delta/2}\binom{N/2-\delta/2}{\delta/2}\binom{N/2-2.\delta/2}{\delta/2}\binom{N/2-3.\delta/2}{\delta/2}\right]}{\sqrt{N}},$$

$$\text{Comb}_{(f,\overline{h})} \simeq \frac{N!}{\left((\delta/2)!\right)^4\left(N-4(\delta/2)\right)!\sqrt{N}}$$

Note that, using $5-$ary polynomials allows to make the combinatorial security of f and g grater than 2^k while choosing for example a large N ($N > 440$ for Type = **A** and $N > 300$ for Type = **B**) and $4\delta \simeq d_f = d_g \simeq 3N/20$ (see example at the end of section 2).

2) With the public key $\overline{h}$, we have $f = g * \overline{h}^{(-1)} \mod q$, and similarly as above, we see that the attacker must perform $\text{Comb}_{(g,\overline{h})} = \text{Comb}_{(f,\overline{h})}$ operations.

Attacks against the one-way function :

The one-way function is:

$\mathcal{S}_{r,d_s} \times \mathcal{S}_{r,d_s} :\xrightarrow{\mathcal{V}} \mathcal{R}_q : (\Gamma_0, \Gamma_1) :\mapsto \mathcal{V}(\Gamma_0, \Gamma_1) = h_0 * \Gamma_0 - h_1 * \Gamma_1 \mod q$

Lattice attack against the one-way function

As in classical NTRU, the lattice parameter is $c_s = \sqrt{\frac{2\pi e||\Gamma_0||.||\Gamma_1||}{Nq}}$. The lattice security must be defined relatively to polynomials which have the smallest norm in $\mathcal{S}_{r,d_s}$ and therefore $c_s \simeq \sqrt{\frac{\pi e \frac{(r-1)}{2} d_s}{Nq}}$.

Meet-in-the-Middle attack against the one-way function.
Since Γ_i is an $r-$ary polynomial, it is not difficult to make the combinatorial security $\text{Comb}_{(\Gamma_i)} \geq \frac{\binom{N/2}{d_s/2}^{(r-1)^{d_s/2}}}{\sqrt{N}}$ grater than 2^k, where k is the security parameter.

2.4 Efficiency

. **Size**

- The public key is $\big((h_0, h_1), (p, c, r, q, d_s)\big)$ and is the private key $\big((f, g, u), (p, c, r, q, d_s)\big)$.
- Then the length of the public key is $\simeq 2N|q| + l$ and the length of the private key is $3N|q| + l$ where l is the length of $d_s||d_u||d_f||d_g||p||c||r||q||N$.
- The length of the message to sign, is $2N$ for **Type** = **A** and $4N$ for **Type** = **B**.
- The signature is $(\Gamma_0, \Gamma_1) \in S_{r,d_s} \times S_{r,d_s}$ then his length is $2N|r|$.

Complexity

1. **Signature** To sign a message $m \in \mathcal{L}_{\text{text}}$, one must compute:
   $\Gamma_0 = f * [u * (\widetilde{m} + \phi) + \psi] \in \mathcal{R}$ and $\Gamma_1 = g * [u * (\phi + c.\varphi) + \psi] \in \mathcal{R}$ with $\widetilde{m} = m$ if $m \in \mathcal{L}_{\text{text}}$
   To compute Γ_0 *and* Γ_1, we need $4N^2$ integer multiplications, $2N$ integer multiplications and $5N$ integer additions,
   Therefore we need $4N^2 + 7N$ elementary operations, but Γ_0 *and* Γ_1 can be computed simultaneously.
   NB1 if $\widetilde{m} = m_1 + pm_2$ with $m = (m_1, m_2) \in \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$, we need $4N^2 + 9N$ elementary operations
   NB2 Note that with fast technics for multiplications we can have a quasi-linear complexity in N.
2. **Verification** similarly as above, here we have $2N^2 + 3N$ elementary operations.
3. **Key generation**: Similarly as above, here we have the number of elementary operations is $\mathcal{O}(N^3)$.

2.5 The role of u and t

. Note that u allows:
- to make difficult transcript attack on Γ_i;
- and to make hight (larger than r) the coefficients of "$h_i\Gamma_i \mod q$" via the presence of u_q; hight coefficients are therefore delated by addition: "$h_0\Gamma_0 + h_1\Gamma_1$

mod q" must have coefficients less than q. In practice, it will suffice, in our particular choice to have, u a $5-ary$ polynomial and d_u a very small integer, for example $d_u \simeq N/100$ with $N > 440$ for Type = **A** and $N > 300$ for Type = **B** (see at the end of the next section).

With the equation $(u * f) * h_0 \mod q = t = (u * g) * h_1 \mod q$, we see that if u is known then one can try to find (f, t) or (g, t) via lattice attacks. Since we had already suggested to choose d_u small, we must choose $d_{\bar{t}}$ hight namely $d_{\bar{t}} = N/2$ because $t = 1 + c\bar{t}$ and $\bar{t}$ is binary.

2.6 Signature and Verification failures

.

Assumption

There exists three negligible functions ν_{cent_q} and ν_{cent_c}, such that for sufficiently large k, we have $\epsilon_{\text{cent}_q} \leq \nu_{\text{cent}_q}(k)$ and $\epsilon_{\text{cent}_c} \leq \nu_{\text{cent}_c}(k)$ where ϵ_{cent_q} and ϵ_{cent_c} are the probabilities :

$\epsilon_{\text{cent}_{r_0}} = \Pr\left[f * [u * (\widetilde{m} + \phi) + \psi] \mod r \neq f * [u * (\widetilde{m} + \phi) + \psi]\right]$
$\epsilon_{\text{cent}_{r_1}} = \Pr\left[g * [u * (\phi + c.\varphi) + \psi] \mod r \neq g * [u * (\phi + c.\varphi) + \psi]\right]$
$\epsilon_{\text{cent}_q} = \Pr\left[t * (\widetilde{m} + c.\varphi) \mod q \neq t * (\widetilde{m} + c.\varphi)\right]$
$\epsilon_{\text{cent}_c} = \Pr\left[\widetilde{m} \mod c \neq \widetilde{m}\right]$

In the next section, we are going to design a security proof using the above probabilities as in NEAP [14]. But to circumvent problems arising with signature and verifications failures, we propose to choose the parameters such that each of those probabilities is zero (justifications of this choice can be fund in APPENDIX B).

Parameters choice (1st step) (See the second step after the security proof at the end of the next section)

Let $p = 2$ and $a = 5$. We have $\epsilon_{\text{cent}_q} = \epsilon_{\text{cent}_c} = \epsilon_{\text{cent}_{r_0}} = \epsilon_{\text{cent}_{r_0}} = 0$ with the following choices.

- **1st case**: $\widetilde{\mathbf{m}} = \mathbf{m}$, we can choose c to be the first integer grater than $p-1 = 1$, then $c = 2$.
  $r_0 = d_g.(a-1)[(p-1).d_u\big(2(p-1)\big) + (p-1)] = 4d_g[2d_u + 1]$,
  $r_1 = d_f.(a-1)[(p-1).d_u\big((p-1) + c(p-1)\big) + (p-1)] = 4d_f[3.d_u + 1]$
  $q_A = (p-1)\big[1 + c(p-1)d_{\bar{t}}\big] = \big[1 + 2d_{\bar{t}}\big]$
  Hence we can choose r respectively q to be the first prime grater than $\max(r_0; r_1)$ respectively $\max(q_A; \alpha.r)$ where α is a security parameter.
- **2nd case**: $\widetilde{\mathbf{m}} = \mathbf{m_1} + \mathbf{pm_2}$, we can choose c to be the first integer grater than $p - 1 + p(p-1) = 3$, then $c = 4$.
  $r_0 = d_g.(a-1)[(p-1).d_u\big(2(p-1) + p(p-1)\big) + (p-1)] = 4d_g[4.d_u + 1]$,
  $r_1 = d_f.(a-1)[(p-1).d_u\big((p-1) + c(p-1)\big) + (p-1)] = 4d_f[5.d_u + 1]$
  $q_B = \big[(p-1) + p(p-1)\big]\big[1 + c(p-1)d_{\bar{t}}\big] = 3.\big[1 + 4d_{\bar{t}}\big]$
  Hence we can choose r respectively q to be the first prime grater than $\max(r_0; r_1)$ respectively $\max(q_B; \alpha.r)$ where α is a security parameter.

3 NSULROM signature: Provable security

For elementary definitions about security proof for signature scheme see "Introduction to Modern Cryptography" [18] of Katz and Lindell, and "Provable Security for Public Key Schemes" of Pointcheval [28] and D. Vergnaud course [32].

3.1 NSULROM: Adaptation to PSS-R model

.

To sign a message $M \in \{0;1\}^{|M|}$ using NSULROM, one uses:

- One compression function: Compress$(x) = (x \mod q)(\mod 2)$;
- Four hash functions H, G, $\overline{H}$ and $\overline{G}$:
  Type = **A**: **signature of a short message**
  Put $N = 1 + |\omega| + |M| + |\rho| = 1 + |\omega| + |\gamma|$, then
  - H maps $|M|+|\rho|$ bits to $|\omega|$ bits and G maps $|\omega|+N$ to $|\gamma| = |M|+|\rho| < N$ bits
  - $\overline{H}$ maps N bits to N bits and $\overline{G}$ maps $N+2$ bits to N bits

  Type = **B**: **signature of a long message** Put $2N = 1 + |\omega| + |M| + |\rho|$, then
  - H maps $|M| + |\rho|$ to $|\omega|$ bits and G maps $|\omega| + N$ to $|\gamma| = |M| + |\rho| > |\omega| + N$ bits
  - $\overline{H}$ maps N bits to $2N$ bits and $\overline{G}$ maps $N+2$ bits to N bits
- Two transformations:
  B2P : $\{0,1\}^N \to \mathcal{L}_{\overline{P},p}$, P2B : $\mathcal{L}_{\overline{P},p} \to \{0,1\}^N$

NB From now, assume that $p = 2$, then B2P is bijective and his reverse is P2B.

To reenforce the security, we propose to define a value π that characterize the scheme i.e a value depending on the public key and the other parameters p, c, r, q, N, Type: for example we can do the following: write $p, c, r, q, N, d_s, d_u, d_f, d_g$ as binary strings. Since these integers are small and $N > 300$, the length l of $d_{\bar{t}}||d_s||d_u||d_f||d_g||p||c||r||q||N$ is less than, there we can pad sufficiently many 0 to have N. Furthermore, we choose 11, to represent **Type** = **A** and 00, to represent **Type** = **B**

- for Type = **A**: computes $\overline{H}(\pi)$ where
  $\pi = \overline{H}(11||d_{\bar{t}}||d_s||d_u||d_f||d_g||p||c||r||q||N||0^{N-l-2}) \oplus \text{P2B}(\text{Compress}(h_0+h_1))$,
- for Type = **B**: defines Trunc_N to be the first N bits, and computes $\overline{H}(\pi)$ where
  $\pi = \text{Trunc}_N[\overline{H}(00||d_{\bar{t}}||d_s||d_u||d_f||d_g||p||c||r||q||N||0^{N-l-2})] \oplus \text{P2B}(\text{Compress}(h_0+h_1))$,

We are going to use π and $\overline{H}(\pi)$ in signature and verification process (but it is optional).

Signature algorithm $S(s_{key}, M)$

To sign a message $M \in \{0,1\}^{|M|}$ with Type **A** or **B** for Ngary, Nague should do the following,

1. obtain his private key $s_{key} = \big((f, g, u); (p, r, c, q)\big)$
2. selects a random $\rho \in \{0,1\}^{|\rho|}$ for padding;
3. computes $\omega = H(M||\rho)$, and sets $\gamma = G(\omega||\pi)$, $S = (M||\rho) \oplus \gamma$;
4. puts $m = \big(0||\omega||S\big) \oplus \overline{H}(\pi)$
   For Type = **A**: since $|m| = N$, defines $\overline{m} = \mathrm{B2P}(m)$.
   For Type = **B** since $|m| = 2N$, splits $m = m_1||m_2$ and defines $\overline{m} = (\overline{m}_1, \overline{m}_2)$ with $\overline{m}_i = \mathrm{B2P}(m_i)$ for $i = 1, 2$.
5. generates randomly $\varphi \in \{0,1\}^N$, puts $\overline{\varphi}_0 = \mathrm{B2P}[\overline{G}(00||\varphi)]$, $\overline{\varphi}_1 = \mathrm{B2P}[\overline{G}(01||\varphi)]$, $\overline{\varphi}_2 = \mathrm{B2P}[\overline{G}(10||\varphi)]$
6. Computes $\Gamma_0 = f*[u*(\widetilde{m} + \overline{\varphi}_0) + \overline{\varphi}_2] \in \mathcal{R}$ and $\Gamma_1 = g*[u*(\overline{\varphi}_0 + c.\overline{\varphi}_1) + \overline{\varphi}_2] \in \mathcal{R}$ with $\widetilde{m} = \overline{m}$ for Type = **A** and $\widetilde{m} = \overline{m}_1 + p\overline{m}_2$ for Type = **B**.
7. if $\Gamma_0 \mod r \neq \Gamma_0$ or $\Gamma_1 \mod r \neq \Gamma_1$ return to (6)
8. if $\min(d_{\Gamma_0}, d_{\Gamma_1}) < d_s$ return to step (6),
9. outputs $\Gamma = (\Gamma_0, \Gamma_1, \mathrm{Type})$

Verification algorithm $V(p_{key})$ (M not given)

1. If $\Gamma = (\Gamma_0, \Gamma_1) \notin \mathcal{S}_{r,d_s} \times \mathcal{S}_{r,d_s}$ rejects the signature.
2. Computes $A = center_q(h_0 * \Gamma_0 - h_1 * \Gamma_1 \mod q)$, where the centering operation $\mathrm{center_q}$ reduces its input into the interval $[\overline{A}; \overline{A} + q - 1]$;
3. Computes $B = center_c(A \mod c)$, where the centering operation $\mathrm{center_c}$ reduces its input into the interval $[\overline{B}; \overline{B} + r - 1]$;
4. For Type = **A** computes $\overline{m} = B \mod p$, and defines $\mathrm{P2B}(\overline{m}) = m$;
   For Type = **B** computes $\overline{m}_1 = B \mod p$, $\overline{m}_2 = (B - \overline{m}_1)/p$, and defines $\mathrm{P2B}(\overline{m}_1) = m_1$, $\mathrm{P2B}(\overline{m}_2) = m_2$, and $m = m_1||m_2$,
5. parse $b_{it}||\omega||S = m \oplus \overline{H}(\pi)$;
6. if $b_{it} = 1$ outputs 0 (= parsing fails);
7. computes $\gamma = G(\omega||\pi)$ and parse $\rho||M = \gamma \oplus S$;
8. if $\omega = H(M||\rho)$, outputs 1 (and M) otherwise outputs 0.

3.2 Security proof

.

NSULROM $-$ $\mathcal{P}$(**k**)**-Problem/Assumption**; we assume that there exists a parameters set $\mathcal{P}(k)$ depending on the security parameter k for which inverting the internal one-way function of NSULROM is hard.

For a proof in the random oracle model, we assume that the attacker can himself compute hash values with the hash functions $\overline{H}$ and $\overline{G}$ but he must queries the simulator when he want to compute hash values with the hash functions H and G.

Theorem 1. .

*- If there exists an attacker $\mathcal{A}$ that $(q_H, q_G, q_{sig}, \tau_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$-solves EUF- CMA(*NSULROM$-$ $\mathcal{P}(k)$*) where $\mathcal{P}(k)$ is a set of parameters for key generation, signing and verification,*

- then there exists a reduction $\mathcal{R}$ simulating the environment of $\mathcal{A}$ in the random oracle model than $(\tau_{\mathcal{B}}, \varepsilon_{\mathcal{B}})$-solves NSULROM$-\mathcal{P}(k)$ *problem with success probability*

$\varepsilon_{\mathcal{B}} \geq \left(1 - (q_{sig} + q_H + 1)\frac{q_G}{2^{N-1-|\omega|}}\right)(1-\frac{1}{2^{|\rho|}}).\mathrm{Pr}_{sol,\mathbf{A}}.(1-\epsilon_{\mathrm{cent}_q}).(1-\epsilon_{\mathrm{cent}_c}).\varepsilon_{\mathcal{A}}$ *if* Type = **A**

$\varepsilon_{\mathcal{B}} \geq \left(1 - (q_{sig} + q_H + 1)\frac{q_G}{2^{N+|\omega|}}\right)(1-\frac{1}{2^{|\rho|}}).\mathrm{Pr}_{sol,\mathbf{B}}.(1-\epsilon_{\mathrm{cent}_q}).(1-\epsilon_{\mathrm{cent}_c}).\varepsilon_{\mathcal{A}}$ *if* Type = **B** *and time bound* $\tau_{\mathcal{B}} \leq \tau_{\mathcal{A}} + q_G q_H \mathcal{O}(1) + q_H \mathcal{O}(1) + (q_H + q_G + q_{sig} + 2)\mathcal{O}(N^2)$ *where:*

- $|\rho|$ *is the size of random used for padding,*
- $\mathcal{R}$ *receives* q_{sig} *signature queries,* q_H *and* q_G *queries respectively for the hash oracles* H *and* G *from* $\mathcal{A}$
- $\epsilon_{\mathrm{cent}_q}$ *and* $\epsilon_{\mathrm{cent}_c}$ *are the probability of "failing the verification process" when the attacker output a valid signature;*
- $\mathrm{Pr}_{sol,\mathbf{A}}$ *and* $\mathrm{Pr}_{sol,\mathbf{B}}$ *are the probability for a element randomly chosen in* R_r *to be in* S_{r,d_s} *(this probability depends on* N *and* d_s *which depends on* Type = **A** *or* **B***);*
- *and* k *is a security parameter.*

Proof For the proof the above Theorem, the reduction algorithm $\mathcal{B}$ and the attacker $\mathcal{A}$ must fix the Type (= **A** or **B**) before beginning the game.

Our reduction $\mathcal{B}$ behaves as follows.

- $\mathcal{B}$ is given $\big(pk = [(h_0, h_1), (p, c, r, q, d_s), \mathrm{Type}] \in \mathcal{P}(k) \leftarrow \mathrm{NSULROM}(1^k)\big)$, $y \leftarrow \mathcal{R}_q$ and a Type, as well as an attacker $\mathcal{A}$ that $(q_T, q_H, q_G, q_T, q_{sig}; \tau_{\mathcal{A}}, \varepsilon_{\mathcal{A}})$-solves EUF- CMA(NSULROM $- \mathcal{P}(k)$) where $\mathcal{P}(k)$ is a set of parameters and k a security parameter;
- $\mathcal{B}$ simulates Gen_{key} and transmits some public key pk to $\mathcal{A}$;
- $\mathcal{B}$ receives signature queries and queries for hash oracles H, G from $\mathcal{A}$
- $\mathcal{A}$ outputs a forgery $\Gamma = (\Gamma_0, \Gamma_1, \mathrm{Type})$ for NSULROM $- \mathcal{P}(k)$;
- $\mathcal{B}$ simulates a verification of the forgery which is valid with probability $\varepsilon_{\mathcal{A}}$;
- $\mathcal{B}$ outputs $(U_0, U_1) \in \mathcal{S}_{r,d_s} \times \mathcal{S}_{r,d_s}$ such that $y = h_0 * U_0 + h_1 * U_1 \mod q$.

Simulation of oracle key generation $\mathbf{G}_{key}$; The reduction $\mathcal{B}$

- Sets $\mathrm{Hist}(S) = \emptyset$ (Signing oracle database);
- sets $\mathrm{Hist}[G] = \emptyset$ (Oracle **G** database);
- sets $\mathrm{Hist}[H] = \emptyset$ (Oracle **H** database);
- sends the NSULROM public key $\big((h_0, h_1), (p, r, c, q)\big)$ to $\mathcal{A}$;
- select an integer K (with $K < 2^{|\rho|}$) and random integers $i_1, ..., i_K \in [1, 2^{|\rho|}]$ where $2^{|\rho|}$ is the number of random to be used for signature process; and put $\Lambda = \{i_1, ..., i_K\}$.

Simulation of oracle hash H; when $\mathcal{A}$ queries H with message, a random and a Type (M, ρ, Type), $\mathcal{B}$,

- checks in $\mathrm{Hist}[H]$, if (M, ρ) was queried in the past. If $H(M||\rho)$ is already defined as $h_{M||\rho}$, returned $h_{M||\rho}$;
- picks the unique j such that $\rho = \rho_j$
- if $j \notin \Lambda$, $\mathcal{B}$;

a) randomly picks $\Gamma_{j,M}^{(0)}, \Gamma_{j,M}^{(1)} \in S_{r,d_s}$, computes $A_{j,M} = h_0 * \Gamma_{j,M}^{(0)} + h_1 * \Gamma_{j,M}^{(1)} \mod q$ and $A_{j,M} = B_{j,M} \mod c$
- for Type = **A**, computes $\overline{m}_{j,M} = B_{j,M} \mod p$, and defines $\text{P2B}(\overline{m}_{j,M}) = m_{j,M}$ and
- for Type = **A** computes $\overline{m}_{j,M}^{(1)} = B_{j,M} \mod p$, $\overline{m}_{j,M}^{(2)} = (B_{j,M} - \overline{m}_{j,M}^{(1)})/p$, and defines $\text{P2B}(\overline{m}_{j,M}^{(1)}) = m_{j,M}^{(1)}$, $\text{P2B}(\overline{m}_{j,M}^{(1)}) = m_{j,M}^{(1)}$, and $m_{j,M} = m_{j,M}^{(1)} || m_{j,M}^{(2)}$,

b) if the first bit of $m_{j,M} \oplus \overline{H}(\pi)$ is not 0, return to (a)
c) parses $m_{j,M} \oplus \overline{H}(\pi) = 0 || \omega_{j,M} || S_{j,M}$;
d) if $(\omega_{j,M} || \pi, \star) \in \text{Hist}[G]$, $\mathcal{B}$ abort;
e) defines and returns $H(M||\rho_j) = \omega_{j,M}$ to $\mathcal{A}$;
f) memorizes $(M, \rho_j, \pi, \overline{H}(\pi), \Gamma_{j,M}^{(0)}, \Gamma_{j,M}^{(1)}, \omega_{j,M}, \text{Type})$ in $Hist[H]$
g) computes $\gamma_{j,M} = (\rho_j || M) \oplus S_{j,M}$
h) memorizes $(\omega_{j,M} || \pi, \gamma_{j,M}, \text{Type}) \in \text{Hist}[G]$
i) outputs $\omega_{j,M}$

– if $j \in \Lambda$, $\mathcal{B}$;

a) randomly picks $\Gamma_{j,M}^{(0)}, \Gamma_{j,M}^{(1)} \in S_{r,d_s}$ and computes $A_{j,M} = y + h_0 * \Gamma_{j,M}^{(0)} + h_1 * \Gamma_{j,M}^{(1)} \mod q$ and $A_{j,M} = B_{j,M} \mod c$
For Type = **A** computes $\overline{m}_{j,M} = B_{j,M} \mod p$, and defines $\text{P2B}(\overline{m}_{j,M}) = m_{j,M}$ and $\text{P2B}(\overline{\lambda}_{j,M}) = \lambda_{j,M}$;
For Type = **B** computes $\overline{m}_{j,M}^{(1)} = B_{j,M} \mod p$, $\overline{m}_{j,M}^{(2)} = (B_{j,M} - \overline{m}_{j,M}^{(1)})/p$, and defines $\text{P2B}(\overline{m}_{j,M}^{(1)}) = m_{j,M}^{(1)}$, $\text{P2B}(\overline{m}_{j,M}^{(1)}) = m_{j,M}^{(1)}$, and $m_{j,M} = m_{j,M}^{(1)} || m_{j,M}^{(2)}$,

b) if the first bit of $m_{j,M} \oplus \overline{H}(\pi)$ is not 0, return to (a);
c) parses $m_{j,M} \oplus \overline{H}(\pi) = 0 || \omega_{j,M} || S_{j,M}$;
d) if $(\omega_{j,M} || \pi, \star) \in \text{Hist}[G]$, $\mathcal{B}$ abort;
e) defines and returns $H(M||\rho_j) = \omega_{j,M}$ to $\mathcal{A}$;
f) memorizes $(M, \rho_j, \pi, \overline{H}(\pi), \perp_0, \perp_1, \omega_{j,M}, \text{Type})$ in $\text{Hist}[H]$;
g) Computes $\gamma_{j,M} = (\rho_j || M) \oplus S_{j,M}$;
h) memorizes $(\omega_{j,M} || \pi, \gamma_{j,M}, \text{Type}) \in \text{Hist}[G]$;
i) outputs $\omega_{j,M}$

Simulation of oracle hash G when $\mathcal{A}$ sends a new query $(\omega||\pi, \text{Type})$, $\mathcal{B}$

– checks in $\text{Hist}[G]$, if $G(\omega||\pi)$ was computed in the past. If $G(\omega||\pi)$ is already defined as $g_{\omega||\pi}$, returned $g_{\omega||\pi}$;
– If not, randomly picks $g_{\omega||\pi} \in \{0,1\}^{|\omega|+|\pi|}$, defines and returns $G(\omega||\pi) = g_{\omega||\pi}$ to $\mathcal{A}$;
– memorizes $\big(\omega||\pi, g_{\omega||\pi}, \text{Type}\big)$ in $\text{Hist}[G]$;
– outputs $g_{\omega||\pi}$.

Simulation of oracle signature $\mathbf{S^{T,H,G}}$; when $\mathcal{A}$ requests the signature of some message M with a Type, $\mathcal{B}$,

- selects randomly i in $[1, 2^{|\rho|}] \setminus \Lambda$,
- invokes its own simulation of H and G to compute $\omega_i = H(M||\rho_i)$;
- search the unique $(M, \rho_j, \pi_{j,M}, \overline{H}(\pi_{j,M}), \beta_1, \beta_2, \omega_{j,M}\text{Type},)$ in Hist$[H]$ and returns $\beta = (\beta_0, \beta_1, \text{Type})$ as signature;
- store $\beta = (\beta_0, \beta_1, \text{Type})$ in oracle database Hist$[S]$.

Simulation of verification $\mathbf{V^{T,H,G}}$; Given a signature $\Gamma = \big((\Gamma_0, \Gamma_1), \text{Type}\big)$, $\mathcal{B}$ do the following, (M not given)

- if $(\Gamma_0, \Gamma_1) \notin S_{r,d_s} \times S_{r,d_s}$ rejects the signature;
- computes $A = center_q(h_0 * \Gamma_0 - h_1 * \Gamma_1 \mod q)$, where the centering operation center$_\text{q}$ reduces its input into the interval $[\overline{A}; \overline{A} + q - 1]$;
- computes $B = center_c(A \mod c)$, where the centering operation center$_\text{c}$ reduces its input into the interval $[\overline{B}; \overline{B} + c - 1]$;
- - for Type = **A** computes $\overline{m} = B \mod p$, and defines P2B$(\overline{m}) = m$;
  - for Type = **B** computes $\overline{m}_1 = B \mod p$, $\overline{m}_2 = (B - \overline{m}_1)/p$, and defines P2B$(\overline{m}_1) = m_1$, P2B$(\overline{m}_2) = m_2$, and $m = m_1||m_2$;
- parses $b_{it}||\omega||S = m \oplus \overline{H}(\pi)$;
- if $b_{it} = 1$ outputs 0 (= parsing fails);
- invokes its own simulation of G, to compute $\gamma = G(\omega||\pi)$ and parse $\rho||M = \gamma \oplus S$;
- invokes its own simulation of H to compute $H(M||\rho)$, if $\omega = H(M||\rho)$, outputs 1 (and M) otherwise outputs 0.

Final Outcome: assume that at the end of the game, $\mathcal{A}$ outputs $\Gamma^* = \big((\Gamma_0^*, \Gamma_1^*), \text{Type}^*\big)$ as a forgery. Then,

- $\mathcal{B}$ simulates $V^{H,G}$ to verify if Γ^* is a valid forgery;
- if center$_\text{q}$ fails, or center$_\text{c}$ fails, $\mathcal{B}$ aborts;
- if the verification returns 0 or $\Gamma^* \in Hist(S)$, $\mathcal{B}$ aborts;
- $\mathcal{B}$ computes $A = center_q(h_0 * \Gamma_0^* - h_1 * \Gamma_1^* \mod q)$ and $B = center_c(A \mod c)$,
- - for Type* = **A**: $\mathcal{B}$ computes $\overline{m} = B \mod p$, and defines P2B$(\overline{m}) = m$
  - for Type* = **B**: $\mathcal{B}$ computes $\overline{m}_1 = B \mod p$, $\overline{m}_2 = (B - \overline{m}_1)/p$, and defines P2B$(\overline{m}_1) = m_1$, P2B$(\overline{m}_2) = m_2$, and $m = m_1||m_2$,
- $\mathcal{B}$ parses $b_{it}||\omega||S = m \oplus \overline{H}(\pi)$;
- $\mathcal{B}$ invokes its own simulation of G, to compute $\gamma = G(\omega||\pi)$ and parse $\rho||M = \gamma \oplus S$;
- $\mathcal{B}$ search for $(M, \rho, \pi, \overline{H}(\pi), \beta_0, \beta_1, \omega, \text{Type}^*) \in$ Hist$[H]$; there exists $j_0 \in [1, 2^{|\rho|}]$ such that $\rho = \rho_{j_0}$;
- if $j_0 \notin \Lambda$, $\mathcal{B}$ aborts;
- $\mathcal{B}$ sets $U_0 = \Gamma_0^* - \beta_0 \mod r$ $U_0 = \Gamma_1^* - \beta_1 \mod r \in \mathcal{R}_r$;
- if $(U_0, U_1) \notin S_{r,d_s} \times S_{r,d_s}$, $\mathcal{B}$ aborts;
- $\mathcal{B}$ outputs $U = (U_0, U_1)$.

Tightness of this reduction:

- $\mathcal{B}$ perfectly simulates oracles key generation, hash function G;

- With Type = **A**, we have $|G(\omega||\pi)| = |\gamma| = |M| + |\rho| \leq |\omega| + |\pi|$, then $\mathcal{B}$ success simulating H with probability grater than $1 - \frac{q_G}{2^{|M|+|\rho|}}$, at each queries.
  With Type = **A**, we have $|G(\omega||\pi)| = |\gamma| = |M| + |\rho| \geq |\omega| + |\pi| = |\omega| + N$, then $\mathcal{B}$ success simulating H with probability grater than $1 - \frac{q_G}{2^{|\omega|+N}}$, at each queries.
- hence, under probability $\left(1 - \frac{q_G}{2^{|M|+|\rho|}}\right)^{q_{sig}+q_H}$ for Type = **A** and under probability $\left(1 - \frac{q_G}{2^{|\omega|+N}}\right)^{q_{sig}+q_H}$ for Type = **B**, the reduction algorithm B reaches the end of the game;
- $\mathcal{A}$ outputs Γ^* with probability at least $\varepsilon_{\mathcal{A}}$; after time $\tau_{\mathcal{A}}$;
- $\mathcal{B}$ success to simulate $V^{H,G}$ with probability;
  $\left(1 - \frac{q_G}{2^{|M|+|\rho|}}\right)(1 - \epsilon_{\text{cent}_q}).(1 - \epsilon_{\text{cent}_c})\varepsilon_{\mathcal{A}}$ for Type = **A** and with probability;
  $\left(1 - \frac{q_G}{2^{|\omega|+N}}\right)(1 - \epsilon_{\text{cent}_q}).(1 - \epsilon_{\text{cent}_c})\varepsilon_{\mathcal{A}}$ for Type = **B**
- since Λ is independent from $\mathcal{A}$, the event $j_0 \in \Lambda$ occurs with probability $\frac{K}{2^{|\rho|}}$. then $\mathcal{B}$, with probability $\frac{K}{2^{|\rho|}}$, outputs a "solution" $U = (U_0, U_1)$ with $U_0 = \Gamma_0^* - \beta_0 \mod r$ and $U_1 = \Gamma_1^* - \beta_1 \mod r \in \mathcal{R}_r$ such that $y = h_0 * U_0 + h_1 * U_1 \mod q$ which is valid with probability $\Pr_{sol,\text{Type}} = Pr\{(U_0, U_1) \in S_{r,d_s} \times S_{r,d_s}\}$

Summing up, $\mathcal{B}$ succeeds with probability

$\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{q_G}{2^{|M|+|\rho|}}\right)^{q_{sig}+q_H+1} \frac{K}{2^{|r|}}.\Pr_{sol,\mathbf{A}}(1-\epsilon_{\text{cent}_q}).(1-\epsilon_{\text{cent}_c}).\varepsilon_{\mathcal{A}}$ for Type = **A**

and $\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{q_G}{2^{|\omega|+N}}\right)^{q_{sig}+q_H+1} \frac{K}{2^{|r|}}.\Pr_{sol,\mathbf{B}}.(1-\epsilon_{\text{cent}_q}).(1-\epsilon_{\text{cent}_c}).\varepsilon_{\mathcal{A}}$ for Type = **B**

and time bound $\tau_{\mathcal{B}} \geq \tau_{\mathcal{A}} + q_G q_H \mathcal{O}(1) + q_H \mathcal{O}(1) + (q_H + q_G + q_{sig} + 2)\mathcal{O}(N^2)$ for Type = **A** or **B**.

- Recall that $U = (U_0, U_1)$ with $U_0 = \Gamma_0^* - \beta_0 \mod r$ and $U_1 = \Gamma_1^* - \beta_1 \mod r \in \mathcal{S}_{r,d_s}$. Since U_0 and U_1 are random polynomials, then $\Pr_{sol,\text{Type}} = \frac{|\mathcal{S}_{r,d_s}|^2}{|R_r|^2} = \frac{\left[\sum_{i=d_s}^{N} \binom{N}{i} (r-1)^i\right]^2}{r^{2N}}$. (Note that we can choose d_s such that this probability is close to 1).

Now, if we choose, the maximal value of $K = 2^{|\rho|} - 1$.

For Type = **A** , since $N - 1 - |\omega| = |M| + |\rho|$ we have

$\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{q_G}{2^{N-1-|\omega|}}\right)^{q_{sig}+q_H+1} (1 - \frac{1}{2^{|\rho|}}).\Pr_{sol,\mathbf{A}}.(1 - \epsilon_{\text{cent}_q}).(1 - \epsilon_{\text{cent}_r}).\varepsilon_{\mathcal{A}}$

therefore $\varepsilon_{\mathcal{B}} \geq \left(1 - (q_{sig} + q_H + 1)\frac{q_G}{2^{N-1-|\omega|}}\right)(1 - \frac{1}{2^{|\rho|}}).\Pr_{sol,\mathbf{A}}.(1-\epsilon_{\text{cent}_q}).(1-\epsilon_{\text{cent}_r}).\varepsilon_{\mathcal{A}}$

For Type = **B**, we have $\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{q_G}{2^{|\omega|+N}}\right)^{q_{sig}+q_H+1} \frac{K}{2^{|r|}}.\Pr_{sol,\mathbf{B}}.(1-\epsilon_{\text{cent}_q}).(1-\epsilon_{\text{cent}_r}).\varepsilon_{\mathcal{A}}$ therefore $\varepsilon_{\mathcal{B}} \geq \left(1 - (q_{sig} + q_H + 1)\frac{q_G}{2^{N+|\omega|}}\right)(1 - \frac{1}{2^{|\rho|}}).\Pr_{sol,\mathbf{B}}.(1-\epsilon_{\text{cent}_q}).(1-\epsilon_{\text{cent}_r}).\varepsilon_{\mathcal{A}}$

3.3 Security level

In this subsection, we study particular cases that allows to choose all the parameters as a function of the parameter security k. We will only consider the cases: $k = 120,\ 160,\ 224$.

We do not guaranty that our approach gives a rigorous method, but it gives general ideas of how we can choose the parameters for a fixed level security which allows:

- to circumvent the most efficient attacks (Transcript Attack, Lattice Attack and Meet-in-the-Middle attack) against the private/public key and the internal one-way function;
- to make tight the security reduction (specially for the success probability).

See in appendix for "How to choose parameters in order to avoid signature and verification failures?".

First of all, we must have $N = \mathcal{O}(k)$ and we assume that if M is the message to sign then $|M| = k$ for short message (Type = **A**) and $|M| = 3k$ for long message (Type = **B**). Since, for the four hash functions H, G, $\overline{H}$ and $\overline{G}$, we need only the randomness and the one-wayness of these functions, it will suffice (for the three levels of security considered here) to choose the hash values such that $|H| = |\omega| = 224$, $|G| = |M| + |\rho|, N + |\omega| > 224$, $|\overline{H}| = N$ and $|\overline{G}| = N$.

For the three levels of security considered here, we can assume that $(q_{sig} + q_H + 1)q_G \leq 2^{120}$ then $(q_{sig} + q_H + 1)\frac{q_G}{2^{N-1-|\omega|}} \leq \frac{1}{2^{N-344}}$ for Type = **A**
and $(q_{sig} + q_H + 1)\frac{q_G}{2^{N+|\omega|}} \leq \frac{1}{2^{N+104}}$ for Type = **B**.

1. For Type = **A**
   a) We have $r_0 = 4d_g[2d_u + 1]$, $r_1 = 4d_f[3.d_u + 1]$ and $q_A = \left[1 + 2d_{\overline{t}}\right]$.
   Hence we can choose r respectively q to be the first prime grater than $\max(r_0, r_1)$ respectively $\max(q_A, r)$ with the order of $q \mod N$ is grater than $\frac{N-1}{2}$.
   and, in those cases $\epsilon_{\text{cent}_c} = \epsilon_{\text{cent}_q} = \epsilon_{\text{cent}_{r_0}} = \epsilon_{\text{cent}_{r_1}} = 0$.
   Therefore, we have $\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{1}{2^{N-344}}\right)\left(1 - \frac{1}{2^{|\rho|}}\right)\text{Pr}_{sol,\mathbf{A}}\varepsilon_{\mathcal{A}}$.
   Now if we put $N - 344 \geq |\rho|$ we have:
   $\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{1}{2^{|\rho|}}\right)^2\text{Pr}_{sol,\mathbf{A}}\varepsilon_{\mathcal{A}} \geq \left(1 - \frac{1}{2^{|\rho|-1}}\right)\text{Pr}_{sol,\mathbf{A}}\varepsilon_{\mathcal{A}}$. Note that if $|\rho| = 16$ then $\varepsilon_{\mathcal{B}} \geq 0,999969.\text{Pr}_{sol,\mathbf{A}}.\varepsilon_{\mathcal{A}}$
   b) For k-bit security, we require to set N to be the first prime greater than $\max(k + 1 + \lambda + 11 + 224,\ \lambda + 11 + 344) = \max(k + \lambda + 236,\ \lambda + 355)$.
2. For Type = **B**
   a) We have $r_0 = 4d_g[4.d_u + 1]$, $r_1 = 4d_f[5.d_u + 1]$ and $q_B = 3.\left[1 + 4d_{\overline{t}}\right]$
   Hence we can choose r respectively q to be the first prime grater than $\max(r_0, r_1)$ respectively $\max(q_B, r)$ with the order of $q \mod N$ is grater than $\frac{N-1}{2}$. and, in those cases $\epsilon_{\text{cent}_q} = \epsilon_{\text{cent}_r} = 0$. Therefore, we have $\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{1}{2^{N+104}}\right)\left(1 - \frac{1}{2^{|\rho|}}\right)\text{Pr}_{sol,\mathbf{B}}\varepsilon_{\mathcal{A}}$. Since $N + 64 \geq |\rho|$ we have:
   $\varepsilon_{\mathcal{B}} \geq \left(1 - \frac{1}{2^{|\rho|}}\right)^2\varepsilon_{\mathcal{A}} \geq \left(1 - \frac{1}{2^{|\rho|-1}}\right)\text{Pr}_{sol,\mathbf{B}}\varepsilon_{\mathcal{A}}$. Note that if $|\rho| = 16$ then $\varepsilon_{\mathcal{B}} \geq 0,999969.\text{Pr}_{sol,\mathbf{B}}.\varepsilon_{\mathcal{A}}$
   b) For k-bit security, we require to set N to be the first prime greater than $\frac{3k+\lambda+236}{2}$.

Corollary 1. *For each $k \geq 120$, there exists a parameters set $\mathcal{P}_0(k)$ such that $\varepsilon_{\mathcal{B}} \geq (1 - \frac{1}{2^{|\rho|-1}}).\Pr_{sol,\text{Type}}.\varepsilon_{\mathcal{A}}$ for* Type = **A** *or* **B**

Security level: If we summarize the above remarks, comments and of results attacks, we can consider the following algorithm for the choice of the parameters: let k be a security parameter and λ be a positive (small) integer (for example $\lambda = \lfloor \log_2 k \rfloor$), to generate the parameters of the scheme, Ngary:

1. initializes $N = 449$ for Type = **A** and $N = 236$ for Type = **B**
2. increments N to the next prime strictly larger than N.
3. chooses $p = 2,\ \ a = 5$, and $c = 2$, for Type = **A** and $p = 2,\ \ a = 5$, $c = 4$, for Type = **B**
4. if $N < \max(k + \lambda + 236,\ \lambda + 355)$, for Type = **A** and $N < \frac{3k+\lambda+236}{2}$ for Type = **B**, then returns to step (2);
5. chooses $d_{\overline{t}} = N/2$, $d_u \simeq N/100$, after chooses the smallest value $d_f = d_g \simeq 4\delta$ for which the combinatorial security values f and g (of section 2) are grater than 2^k;
6. - for Type = **A**, computes $r_1 = 4d_f[3.d_u + 1]$ and $q_A = \left[1 + 2d_{\overline{t}}\right]$; chooses r respectively q to be the first prime grater than r_1 respectively $\max(q_A, \alpha.r)$ with α a security parameter (in general $\alpha = 1,5$ or $\alpha = 2$ will suffice).
   - for Type = **B**, computes $r_1 = 4d_f[5.d_u + 1]$ and $q_B = 3.\left[1 + 4d_{\overline{t}}\right]$ and chooses r respectively q to be the first prime grater than r_1 respectively $\max(q_B, \alpha.r)$ with α a security parameter (in general $\alpha = 1,5$ or $\alpha = 2$ will suffice).
7. (knowing r), chooses the smallest value d_s, for which the combinatorial security values (of section 2) with the one-way function are grater than 2^k and the probabilities
   $\Pr_{sol,\text{Type}} = \dfrac{\left[\sum_{i=d_s}^{N} \binom{N}{i} (r-1)^i\right]^2}{r^{2N}}$ are close to 1;
8. outputs the parameters of the system:
   $p = 2,\ \ a = 5,\ \ c$, N, $d_f = d_g$, $d_{\overline{t}}$, d_u, r, q, d_s, $c_{\overline{h}}$, c_s and Type

Example of parameters with $\lambda = \lfloor \log_2 k \rfloor$ and $\alpha = 1,5$ in the following table where $d_g = d_f = 4\delta \leq 3N/20$, and $d_u \simeq N/100$, $d_s \leq N/20$, $d_{\overline{t}} \simeq N/2$.

Type = **A**

k	N	r	q	$\lvert M\rvert$	$\lvert\rho\rvert$	$d_g = d_f \simeq 4\delta$	d_u	$d_{\overline{t}}$	d_s	$\lvert H\rvert$	$\lvert G\rvert$
120	449	2081	3137	120	124	40	4	184	18	224	244
160	449	3271	4909	160	84	48	5	205	22	224	244
224	479	5477	8219	224	30	72	6	240	26	224	254

Type = **B**

k	N	r	q	$\lvert M\rvert$	$\lvert\rho\rvert$	$d_g = d_f \simeq 4\delta$	d_u	$d_{\overline{t}}$	d_s	$\lvert H\rvert$	$\lvert G\rvert$
120	301	3361	5051	360	17	40	4	151	18	224	377
160	367	4993	7499	480	29	48	5	164	22	224	509
224	461	8929	13397	672	25	72	6	255	26	224	697

Conclusion

In this paper we have successfully proposed a new signature scheme based on quotient ring and lattices with a security proof. In a further work, it will be interesting:

- to study more deeply the role of u and t and the impact of the value of q/r on the security of the scheme, (in the last examples of this paper, we assume that $q/r = 1,5$ will suffice to make hard to invert the internal one-way function);
- to understand more deeply the one-wayness of the "one-way function";
- to see, if vulnerabilities can arise from the choice of the random ϕ, ψ and φ in the signature process, even if they are generated via the hash function $\overline{G}$;
- to design a more optimal algorithm for security and efficiency in order to generate all required parameters for NSULROM signature;
- and to improve the reduction proposed in this paper for the time and the success probability.

References

1. M. Bellare and P. Rogaway, *Random oracles are practical: a paradigm for designing efficient protocols*. Proceedings of the First Annual Conference on Computer and Communications Security, ACM, 1993.
2. X Boyen: *Lattice Mixing and Vanishing Trapdoors: A Framework for Fully Secure Short Signatures and More*. Public Key Cryptography 2010: 499-517
3. R. Canetti, O. Goldreich and S. Halevi, *The random oracle methodology*, revisited, STOC'98, ACM, 1998.
4. D Cash, D Hofheinz, E Kiltz, C Peikert: *Bonsai Trees, or How to Delegate a Lattice Basis*. EUROCRYPT 2010: 523-552
5. D. Coppersmith and A. Shamir. *Lattice attacks on NTRU*, In Advances in cryptology EUROCRYPT 97, volume 1233 of Lecture Notes in Comput. Sci., pages 52U61. Springer, Berlin, 1997.
6. J.S. Coron, *Optimal security proofs for pss and other signature schemes*, Proceedings of Eurocrypt'02, lncs, vol. 2332, Springer-Verlag, 2002, pp. 272-287.
7. C. Gentry, J. Jonsson, J. Stern, and M. Szydlo *Cryptanalysis of the NTRU Signature Scheme (NSS) from Eurocrypt 2001* available online
8. C. Gentry, M Szydlo, *Cryptanalysis of the Revised NTRU SignatureScheme*, Advances in Cryptology— Eurocrypt '02, Lecture Notes in Computer Science, Springer-Verlag, 2002.
9. . S. Goldwasser, S. Micali and R. Rivest, *A digital signature scheme secure against adaptive chosen-message attacks*, SIAM Journal of computing, 17(2), pp. 281-308, April 1988.
10. S. hasegawa, S. Isobe, M. Mambo, H. Shizuya, Y. Futa and M Ohmori *A contremeasure for protecting NTRUsign agianst the Transcript attak*, Interdisciplinary information sciences, Vol.13, No 2 pp. 181-188 (2007).
11. J. Hoffstein, J. Pipher, and J. H. Silverman. *NTRU :A Ring Based Public Key Cryptosystem in Algorithmic Number Theory*, Lecture Notes in Computer Science 1423, Springer-Verlag, pages 267-288, 1998.
12. J. Hoffstein, J. Pipher, J.H. Silverman. *NSS: The NTRU Signature Scheme*. In Proc. of Eurocrypt '01, LNCS 2045, pages 211-228. Springer-Verlag, 2001.

13. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. Silverman, W. Whyte, *NTRUSign: Digital Signatures Using the NTRU Lattice*, CT-RSA 2003,
14. N. Howgrave-Graham, J. Silverman, A. Singer, and W. Whyte. *NAEP: Provable Security in the Presence of Decryption Failures*. NTRU Cryptosystems. available at http://www.ntru.com.
15. N. Howgrave-Graham, J. Silverman, and M. Whyte. *Choosing Parameter Sets for NTRUEncrypt with NAEP and SVES-3*. 2005. available at http://www.ntru.com.
16. N. Howgrave-Graham, J. Silverman, A. Singer, and W. Whyte. *Meet-in-the-Middle attack on a NTRU private key* available at http://www.ntru.com/cryptolab/tech notes.htm # 004!.
17. Howgrave-Graham, N., Nguyen, P.Q., Pointcheval, D., Proos, J., Silverman, J.H., Singer, A., Whyte, W.: *The Impact of Decryption Failures on the Security of NTRU Encryption*. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 226-246. Springer, Heidelberg (2003)
18. Jonathan Katz, Yehuda Lindell *Introduction to Modern Cryptography*
19. V. Lyubashevsky *Lattice Signatures Without Trapdoors*. IACR cryptology ePrint Archive 2011: 537 (2011)
20. V. Lyubashevsky *Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures*. ASIACRYPT 2009: 598-616
21. P.Q. Nguyen and J. Stern. *The Two Faces of Lattices in Cryptology*. In Proceeding of CaCL'01, LNCS, vol. 2146, Springer-Verlag, pp. 148-180, 2001.
22. NTRU Inc. http://www.ntru.com/
23. www.ntru.com
24. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
25. P. Nguyen *A note on the security of NTRUsign* available at www.eprint.iacr.org/2006/387.pdf
26. P. Nguyen and Oded Regev *Learning a parallelepiped: Cryptanalyse of GGH and NTRUsign* available at www.cs.tau.ac.il/ odedr/papers/gghattack.pdf
27. D. Pointcheval *Provable Security for Public Key Schemes* Advanced Course on Contemporary Cryptology, pages 133-189, June 2005. http://www.di.ens.fr/ pointche/pub.php?reference=Po04
28. D. Pointcheval and J. Stern *Security Proofs for Signature Schemes* Advances in Cryptology Proceedings of EUROCRYPT '96 (may 12-16, 1996, Zaragoza, Spain) U. Maurer, Ed. Springer-Verlag, LNCS 1070, pages 387-398.
29. M Ruckert: *Strongly Unforgeable Signatures and Hierarchical Identity-Based Signatures from Lattices without Random Oracles*. PQCrypto 2010: 182-200
30. J. H. Silverman. *Invertibility in Truncated Polynomial Rings*. Technical Report No. 9. available at http://www.ntru.com.
31. J. H. Silverman and M. Whyte. *Estimating Decryption Failure Probabilities for NTRUEncrypt*. NTRU Cryptosystems technical report No 18. available at http://www.ntru.com.
32. D. Vergnaud *Course "Provable Security in Public-Key Cryptography"* Lecture 1, 2, 3 and 4 in June 2010 at Dakar available at http://www.di.ens.fr/ damien (Ecole Normale Superieure, C.N.R.S. I.N.R.I.A. (France))

APPENDIX

How to choose parameters in order to avoid signature and verification failures

1. We have $\max\big(\text{Width}(\widetilde{m})\big) \leq (p-1)$, if $\widetilde{m} = m$ with $m \in \mathcal{L}_{text,p}$ and $\max\big(\text{Width}(\widetilde{m})\big) \leq (p-1) + p(p-1)$ if $\widetilde{m} = m_1 + pm_2$ with $m = (m_1, m_2) \in \mathcal{L}_{\text{text}} \times \mathcal{L}_{\text{text}}$.
   Therefore, if we choose $(p-1) < c$ for $\widetilde{m} = m$ and $(p-1) + p(p-1) < c$ for $\widetilde{m} = m_1 + pm_2+$, then surely $\epsilon_{\text{cent}_c} = 0$
2. For a polynomial $\overline{P} = (\overline{P}_0, \ldots, \overline{P}_{N-1})$, defines $T_{\overline{P}} = (T_{\overline{P},0}, \ldots, T_{\overline{P},N-1})$ and where $T_{\overline{P},i} = 1$ if $\overline{T}_i \neq 0$ and $T_{\overline{P},0} = 0$ if $\overline{T}_i = 0$. We have $\text{Width}\big(\overline{P} * \overline{Q} \mod q\big) \leq \text{Width}(\overline{P})\text{Width}(\overline{Q})\text{Width}\big(T_{\overline{P}} * T_{\overline{Q}}\big)$
   By choice, we put $\text{Width}(u) = \text{Width}(\bar{t}) = p-1$ and

$$\text{Width}(f) = \text{Width}(g) = \text{Width}(u) = \text{Width}(\bar{t}) = a-1, \text{ with } a = 5 \text{ or } 3$$

   Let $\overline{P} = \widetilde{m} + c\varphi$, we have $\max\big(\text{Wigth}(\overline{P})\big) \leq \max\big(\text{Wigth}(\widetilde{m})\big) + c(p-1)$.
   Since $(\widetilde{m} + c.\varphi) * t = \overline{P} * t = \overline{P} + c\overline{P} * \bar{t}$, we have

$$\max \text{Width}\big(\overline{P} * \bar{t}\big) \leq \max\big(\text{Width}(\overline{P})\big) \max \text{Width}(\bar{t}) \max\big(\text{Width}(T_{\overline{P}} * T_{\bar{t}})\big)$$

   Now using result on "Width of product of binary polynomials" from [15], we have $\max\big(\text{Width}(T_{\overline{P}} * T_{\bar{t}})\big) \leq d_{\bar{t}}$. Hence,

$$\max \text{Width}\big((\widetilde{m}+c.\varphi)*t\big) \leq \max\big(\text{Width}(\widetilde{m}+c.\varphi)\big)+c \max\big(\text{Width}(\widetilde{m}+c.\varphi)\big) \max \text{Width}(\bar{t}) d_{\bar{t}}$$

   Hence,
   $\max \text{Width}\big((\widetilde{m} + c.\varphi) * t\big) \leq (p-1) + (p-1)c(p-1)d_{\bar{t}}$ for $\widetilde{m} = m$ and
   $\max \text{Width}\big[(\widetilde{m} + c.\varphi) * t\big] \leq \big[(p-1) + p(p-1)\big] + \big[(p-1) + p(p-1)\big]c(p-1)d_{\bar{t}}$ for $\widetilde{m} = m_1 + pm_2$
   Therefore, if we choose
   $\boxed{q > q_A = (p-1)\big[1 + c(a-1)d_{\bar{t}}\big]}$ for $\widetilde{m} = m$ and
   $\boxed{q > q_B = \big[(p-1) + p(p-1)\big]\big[1 + c(p-1)d_{\bar{t}}\big]}$ for $\widetilde{m} = m_1 + pm_2$ then surely $\epsilon_{\text{cent}_q} = 0$
3. We have also $\max\big(\text{Wigth}[u * (\widetilde{m} + \phi)]\big) \leq (p-1).d_u[2(p-1)]$ for $\widetilde{m} = m$ and $\max\big(\text{Wigth}[u * (\widetilde{m} + \phi)]\big) \leq (p-1).d_u[2(p-1) + p(p-1)]$ for $\widetilde{m} = m_1 + pm_2$
   We have $\max\big(\text{Wigth}[g*\big(u*(\widetilde{m}+\phi)+\psi\big)]\big) \leq d_g.(a-1)[(p-1).d_u\big(2(p-1)\big)+(p-1)]$ for $\widetilde{m} = m$ and
   $\max\big(\text{Wigth}[g*\big(u*(\widetilde{m}+\phi)+\psi\big)]\big) \leq d_g.(a-1)[(p-1).d_u\big(2(p-1)+p(p-1)\big)+(p-1)]$ for $\widetilde{m} = m_1 + pm_2$
   Similarly, we have $\max\big(\text{Wigth}[f * \big(u * (\phi + c\varphi) + \psi\big)]\big) \leq d_f(a-1)[(p-1)d_u\big((p-1) + c(p-1)\big) + (p-1)]$ for $\widetilde{m} = m$ and
   $\max\big(\text{Wigth}[f*\big(u*(\phi+c\varphi)+\psi\big)]\big) \leq d_f(a-1)[(p-1)d_u\big((p-1)+c(p-1)\big)+(p-1)]$ for $\widetilde{m} = m_1 + pm_2$.
   Now, if we choose $r > \max(r_0, r_1)$ where
   $\boxed{r_0 = d_g.(a-1)[(p-1).d_u\big(2(p-1)\big) + (p-1)]}$
   and
   $\boxed{r_1 = d_f.(a-1)[(p-1).d_u\big((p-1) + c(p-1)\big) + (p-1)]}$
   for $\widetilde{m} = m$ and
   $\boxed{r_0 = d_g.(a-1)[(p-1).d_u\big(2(p-1) + p(p-1)\big) + (p-1)]}$
   and
   $\boxed{r_1 = d_f.(a-1)[(p-1).d_u\big((p-1) + c(p-1)\big) + (p-1)]}$
   for $\widetilde{m} = m_1 + pm_2$,
   then surely $\epsilon_{\text{cent}_{r_0}} = \epsilon_{\text{cent}_{r_1}} = 0$