A lattice based authentication for low-cost RFID

Ethmane El Moustaine, Maryline Laurent

Institut Mines-Télécom, Télécom SudParis, CNRS Samovar UMR 5157
9 rue Charles Fourier, 91011 Evry, France
Email: {Ethmane.ElMoustaine, Maryline.Laurent}@telecom-sudparis.eu

***Abstract*—Security and privacy are major threats for the RFID technology today. First, the RFID passive tags prevailing in most of the RFID applications are very limited in processing power, so they cannot perform complex computations. Second, there are no RFID authentication protocols that can support both scalability and privacy at low complexity cost.**
In this paper, we present an adaptation of NTRU public cryptosystem for low-cost RFID tags and new efficient asymmetrical challenge/response RFID mutual authentication protocol for low-cost RFID systems based on this NTRU's adaptation. Thanks to properties of the polynomial ring in which NTRU operates we have ensured that the tag encrypts challenges using only addition and right circular shifts. The proposed authentication protocol guarantees privacy, high scalability level and low implementation complexity. It takes advantages of NTRU and HMAC features, and is resistant to all the classical security attacks including replays, tracking, man in the middle attacks, etc.

Keywords: Lightweight Cryptography, RFID, Security, Privacy, Scalability, NTRU public key cryptosystem, Mutual authentication

I. Introduction

As Mark Weiser already predicted in 1991, one of the main problems against adoption of ubiquitous computing is privacy [1]. The simultaneous provision of privacy, scalability, and security in low-cost RFID systems does not give other alternatives than designing a lightweight approach based on public keys cryptography.
Public key cryptography can achieve a higher security level compared to symmetric key cryptography, while it requires a higher computational overhead [2].
In this paper, we propose a new efficient asymmetrical mutual authentication protocol especially designed for low-cost RFID systems and based on adaptation that we have introduced on NTRU public key cryptosystem on the tag side. It is demonstrated to be robust against the classical security and privacy attacks performed over the wireless channel. To the best of our knowledge, our protocol is the first solution based on adaptation of NTRU for low-cost RFID tags.
This paper is organized as follows. Section II introduces works related to RFID security issues. Section III gives a brief description of lattice and NTRU cryptosystem. Section IV then describes our asymmetrical mutual authentication protocol and section V discusses its robustness to security and privacy threats. Performance issues aspects are also given in section VI before our conclusions in section VII.

II. Related Works

The reduction of tag's cost is a key factor if a large number of tags are needed like in the supply chain area. As such, a special care is needed when designing security solutions for consuming as fewer resources as possible, especially within the tag.
A variety of protocols based on hash functions, or lightweight cryptography (e.g. XOR) have been proposed to support the authentication service and solve some of the security and privacy issues of the RFID low-cost systems. To counteract clandestine tracking, some of them are doing key update after each successful authentication session so some randomness is introduced into the message content from one session to another.
For privacy reasons, the tag must be untraceable between two legitimate authentication sessions. It is this condition that will complicate the design of RFID authentication protocols.
In [3], *Ohkubo et al.* propose a hash chain approach based on two different hash functions. This protocol does not provide mutual authentication - namely, the reader authenticates the tag while the tag does not authenticate the reader - and it is vulnerable to impersonation attacks and replay attacks. In addition, there are two additional shortcomings in this protocol. One of them is unscalability as the back-end executes an exhaustive search. The other one is the cost of the tag where two different hash functions need to be implemented.
All the above protocols are symmetrical, and as such, they cannot support both privacy and scalability at a moderate operational overhead. Indeed, the back-end, when authenticating a tag, has to do: an exhaustive search (in the keys and identifiers of tags) or to use a pseudo-index sent by the tag (unchangeable between two legitimate authentications) that makes the tag becomes traceable between two key updates.
The asymmetric authentication approaches have not been much addressed by researchers. Only few asymmetrical protocols have been proposed in the literature. *Kaya et al.* propose in [4] an approach for multi-context RFID based on

This work was partially funded by the German Federal Ministry of Education and Research under the promotional reference 01ISO7009 and by the French Ministry of Research within the RESCUE-IT project.

public key cryptography with a case study of NTRU, but without adapting these cryptosystems to RFID tags.
Batina et al. [5], *Kaya et al.* [4], *Lee et al.* [6] propose RFID authentication protocols based on elliptic curve cryptography, but they require the tag to implement scalar point multiplications in the tag.
In [7], *Sekino et al.* propose an authentication protocol based on the Niederreiter public key cryptosystem [8]. It is not adapted to low-cost tags, as the tag is required to support hash function, to store large matrix, and to excute matrix operations.
In [9], *Girault* proposes a storage-computation trade-off approach of the famous GPS scheme [10]. This approach consists in storing t *coupons* of pairs of numbers r_i, x_i (or in storing only x_i if r_i is regenerated on demand locally) for $0 \leq i \leq t-1$ on the tag with the tag-specific secret key s, the computation by the tag is limited to $y = r_i + s \times c$ where c is a challenge generated by the reader. This approach has been widely studied in the literature and different implementations have been proposed [11], [12], [13].
However, this approach does not support a moderate security level. Indeed, an attacker can make a denial of service attack by interrogating the tag more than t times because the number of coupons stored on the tag is limited for cost reasons(storage capacity is the most expensive part of the hardware).

III. Lattice and NTRU cryptosystem

A. Lattice theory

If v_1, v_2,..., v_k are linearly independent vectors of $\mathbb{R}^n$, then a lattice L generated by v_1, v_2,..., v_k is the set:

$$L = \{\sum_{i=1}^{k} a_i v_i \mid \forall i \in \{1,...,k\},\ a_i \in \mathbb{Z}\}$$

The Shortest Vector Problem (SVP): Given a base B (set of vectors linearly independent) of a lattice L, find the smallest possible nonzero vector of L, i.e. find $v \neq 0$ such that $||v||$ is minimal. Where $||.||$ is the euclidean norm, if x=(x_1,...,x_n), then $||x|| = \sqrt{<x,x>} = \sqrt{\sum_{i=1}^{n} x_i^2}$.
The Closest Vector Problem (CVP): Given a base B of a lattice L and a vector $w \notin L$, find a vector $v \in L$ that is the closest to w, i.e. $||v - w||$ is minimal.
The SVP and CVP problems are known as NP-hard.

B. NTRU cryptosystem

NTRU is a probabilistic public key cryptosystem proposed by *Hoffstein et al.* [14], it is considered as secure by the standards IEEE 1363.1 [15] and X9 [16]. NTRU is one of the fastest public key cryptosystems, it is suitable for systems which can not be easily updated as designed for long-term data protection. NTRU operations are performed in the ring $\Re = \mathbb{Z}[X]/(X^N - 1)$, where N is a positive prime, defining the degree of the polynomial, i.e. the base ring $\Re$. NTRU depends on three integer parameters (N, p, q) and four sets D_f, D_g, D_r, D_m of polynomials of degree at most $N-1$. In the latest version of NTRU, $p = 2$ and $gcd(p,q) = 1$, D_m is the space of plaintext with coefficients taken $mod\ p$.

1) Key generation: Several steps are necessary: first choose randomly two polynomials f and g in D_f and D_g, respectively; then inverse f $(mod\ q)$ to obtain f_q and inverse f $(mod\ p)$ to obtain f_p; finally, calculate the public key $h = p{*}g{*}f_q\ (mod\ q)$, the corresponding private key is $(g,\ f)$. The selection of parameters is determining the security level of NTRU. Along the paper, we consider $p = 2$, like in the latest version of NTRU.

2) Encryption: Fast encryption of a message with NTRU includes three operations: Transform the message m to a polynomial of D_m, randomly choose a polynomial $r \in D_r$ and calculate the cipher $e = r{*}h + m\ (mod\ q)$.

3) Decryption: Decryption is processed as follows: Use the private key $(g,\ f)$, calculate $a = f{*}e\ (mod\ q)$, choose the coefficients of a in the interval from -q/2 to q/2, finaly retrieve the message by computing $f_q{*}a\ (mod\ p)$.
The NTRU problems for breaking the private key or the decryption algorithm are known as the NP-hard SVP and CVP problems respectively, in the NTRU's lattice $L_{NTRU} = \{(u,v) \in \Re^2 \mid v{*}h - u = 0\ (mod\ q)\}$.

IV. The proposed protocol

Our protocol is an asymmetrical probabilistic mutual authentication based on the NTRU public key cryptosystem [14]. *Atici et al.* have designed in [17] an NTRU's architecture for encryption/decryption that requires only 10.8 *kgate* for $(N,p,q) = (167,3,128)$, our solution does not dedicate such amount of resources for the classical operations of NTRU on the tag, because all complex operations of NTRU such as modular arithmetic, polynomials multiplication are done at the server while the tag implements only lightweight operations. To the best of our knowledge, our protocol is the first lightweight RFID authentication protocol for low-cost RFID tags that simultaneously supports privacy (tracking resistance), high scalability level, and security protection at a moderate operational overhead, i.e. in terms of tag's capacity, exhaustive search and storage capacity at the back-end.

A. Our adaptation of NTRU to low-cost RFID tags

The public key cryptosystem NTRU handles polynomials in the ring $\Re = \mathbb{Z}[X]/(X^N - 1)$ consequently, all the polynomials are of degree less than N.
Any element f of $\Re$ is represented by:

$$f = (f_0, f_1, ..., f_{N-1}) = \sum_{i=0}^{N-1} f_i x^i$$

Multiplication in the ring $\Re$ is a convolution $mod\ X^N - 1$. In $\Re$, the addition of two polynomials is done term by term. However, a multiplication by X is equivalent to rotate the coefficients: the coefficient of X^i becomes the coefficient of X^{i+1} (and since multiplications are $mod\ X^N - 1$, the monomial X^N is equal to 1). Specifically, this convolution works as follows:
Let's $(f,g) \in\ \Re^2$ the convolution product defined by:

$(f{*}g)(X) = \sum_{i=0}^{N-1} h_i X^i$, $h_k = \sum_{j=0}^{k} f_j g_{k-j} + \sum_{j=k+1}^{N-1} f_j g_{N+k-j}$

If $f^{'}$ is a right circular shift of f by i-position, the product $f^{'}{*}g$ is the right circular shift of $f{*}g$ by i-position. Indeed, a right circular shift of f by i-position is exactly equal to $rot_i(f) = X^i{*}f\ mod\ (X^N - 1)$ where $X^N{*}f\ mod\ (X^N - 1) = f$. So $f^{'}{*}g = X^i{*}f{*}g\ mod\ (X^N - 1) = X^i{*}(f{*}g)\ mod\ (X^N - 1) = rot_i(f{*}g)$, then:

$$rot_i(f{*}g) = rot_i(f){*}g \qquad (1)$$

If we add $f{*}g$ to its right-shift rotation by i-position (1), we obtain: $f{*}g + rot_i(f{*}g) = f{*}g + X^i{*}f{*}g\ mod\ (X^N - 1) = (f + X^i{*}f){*}g\ mod\ (X^N - 1) = (f + rot_i(f)){*}g$, then:

$$f{*}g + rot_i(f{*}g) = (f + rot_i(f)){*}g \qquad (2)$$

The largest polynomial's coefficient handled along this paper is equal to $2q - 1$, for this we choose to write each polynomial's coefficient in $\lceil log_2(2q-1)\rceil$ bits to facilitate the manipulation of polynomials on the tag side. Then, each rotation of $r{*}h$ by $s - bit$, where s is a multiple of $\lceil log_2(2q-1)\rceil$, corresponds to the new product $r^{'}{*}h$, where $r^{'}$ is a right circular shift of r by $s - bit$ (cf. Equation (1)), and each $r{*}h + rot_s(r{*}h)$ corresponds to another product $r_s{*}h$ where $r_s = r + r^{'}$ (cf. Equation (2)). So, if the tag has in memory $r{*}h\ (mod\ q)$, it can construct easily an $r_s{*}h$ and encrypts a challenge m by adding it to $r_s{*}h$. This point will be later used in this paper.
Note that, in $NTRU$ specifications, r is any small polynomial, its coefficients can be chosen in {-1, 0, 1} or $\{0, 1\}$ to simplify the implementation or to attain very high security level according to the implementation standard [18], [19]. In our approach, few coefficients of this polynomial (r_s) can be equal to 2 or -2, such modification does not have a significant impact on the security of $NTRU$. Indeed, breaking the private key is an SVP problem in the NTRU's lattice $L_{NTRU} = \{(u, v) \in \Re^2 \mid v{*}h - u = 0\ (modq)\}$ which is independent of the choice of r. On the other hand, the decryption problem can be seen as a CVP problem: the cipher of a message m is $e = r{*}h + m\ (mod\ q)$; this means that the vector $(e, 0)$ is close to the vector $(pr{*}h\ mod\ q, pr)$ of the NTRU's lattice. More precisely, the difference between the two vectors is (m, pr) which is by definition very short. Consequently, if few coefficients of r are equal to 2 and/or -2, the difference (m, pr) between these two vectors remains very short. Then, for an attacker, decrypting a ciphertext with no knowledge of the private key is still a difficult CVP problem in the NTRU's lattice.

B. Initialization

Each tag is initialized with a random long-term secret key k_t and two secret polynomials (binary vector): an identity $id \in D_m$ that is unique at the back-end, and a polynomial $r{*}h$ calculated modulo q in $\Re$ where r is randomly generated in D_r. The back-end stores only one private key $(g,\ f)$ to authenticate all tags, and an id for each tag.

$r{*}h\ (mod\ q)$	Designed by $r{*}h$
$r^{'}{*}h\ (mod\ q)$	Designed by $r^{'}{*}h$
$\oplus$	Exclusive-or operator
$HW(x)$	Hamming weight of x
$rot(x, y)$	Right circular shift over x by $HW(y)$
$rot^{-1}(x, y)$	Left circular shift over x by $HW(y)$
$rot(x, K'y)$	Right circular shift over x by K times the $HW(y)$
$\lceil x\rceil$	Smallest integer not less than x
$H_{k_t}()$	Hash-based Message Authentication Code (HMAC)
k_t	long-term secret key
$x_{[0,s-1]}$	the s-least significant bits of x

TABLE I: Notations.

C. Description

As described in figure 1, based on the notations given in Table I, when the tag is queried, it reads the pre-computed value M, it calculates $M = H_{k_t}(M)$, $r_t = M_{[0,s-1]}$ and it constructs a n-bit binary vector (polynomial) $m = B2P(M)$, where $B2P : \{0,1\}^l \to D_m$ converts a sequence of bits into a binary polynomial, where $n = \lceil log_2(2q-1)\rceil N$. Note that, $m \in D_m$ as each of its coefficients is in $\{0, 1\}$. Then the tag reads the pre-computed value $r{*}h$ and calculates $r_s{*}h = r{*}h + rot(r{*}h, \lceil log_2(2q-1)\rceil' r_t)$ and replaces $r{*}h$ and M with $rot(r{*}h, \lceil log_2(2q-1)\rceil' r_t)$ and $H_{k_t}(M)$ respectively. After that, the tag calculates and sends back to reader $e_1 = r_s{*}h{+}m$. Note that, if the previous authentication attempt has been successfully achieved, in its polynomial form, e_1 can be written as $e_1 = r_s{*}h + m = (X^{HW(r_t)} + 1){*}r{*}h + m\ mod(X^N - 1)$ (cf. section IV-A). If there is one or more failed previous authentication attempts, in its polynomial form, e_1 can be written:

$$e_1 = (X^i + X^j){*}r{*}h + m\ mod(X^N - 1), (i, j) \in\ \mathbb{N}^2 \qquad (3)$$

As such, the tag only does two sum operations of four binary values, few xor operations, some right circular shifts, one HMAC.

Upon receiving the tag's response (e_1), the reader/back-end calculates and decrypts $e_1\ (mod\ q)$ to retrieve m. Note that, $e_1\ (mod\ q)$ is a valid NTRU ciphertext. That is, $e_1\ (mod\ q)$ is computed by the reader as each coefficient of e_1 is at most equal to $2q - 1$: each coefficient of m is in $\{0, 1\}$ and each coefficient of $r_s{*}h$ used in the computing of e_1 is between 0 and $2q - 1$ (because each coefficient of $r{*}h$ is between 0 and q-1). The reader/back-end decrypts $e_1\ (mod\ q)$ using the secret key $(g,\ f)$ to retrieve m. It then generates randomly a polynomial ($n-$bit pseudo-random sequence) $r^{'} \in D_r$ and computes $r^{'}{*}h\ (mod\ q)$ in $\Re$, $e_2 = rot(r_s{*}h, r_s{*}h) \oplus (r^{'}{*}h)$, and $e_3 = H_M(r^{'}{*}h)$. Then the reader/back-end sends these values to the tag. Upon receiving e_2 the tag retrieves $r^{'}{*}h$ as $r^{'}{*}h = e_2 \oplus rot(r_s{*}h, r_s{*}h)$, it authenticates the back-end/reader by checking the correctness of e_3, which authenticates $r^{'}{*}h$, it can be generated only by the legitimate reader thanks to HMAC. If e_3 is correct the reader/back-end is authenticated. Otherwise the tag aborts the session concluding that the reader failed to decrypt e_1 correctly.
If reader/back-end is authenticated, the tag sends back

Reader/back-end		Tag
	$\xrightarrow{hello}$	find $r{*}h$, k_t, M, compute $M \leftarrow H_{k_t}(M)$, $m = B2P(M)$, $r_t = M_{[0,s-1]}$, $r_s{*}h = r{*}h + rot(r{*}h, \lceil log_2(2q-1)\rceil' r_t)$
decrypt e_1 $(mod\ q)$, get m	$\xleftarrow{e_1}$	$e_1 = r_s{*}h{+}m$, $r{*}h \leftarrow rot(r{*}h, \lceil log_2(2q-1)\rceil' r_t)$
pick a random $r^{'} \in D_r$, compute $e_2 = rot(r_s{*}h, r_s{*}h) \oplus (r^{'}{*}h)$ and $e_3 = H_M(r^{'}{*}h)$	$\xrightarrow{e_2, e_3}$	get $r^{'}{*}h = e_2 \oplus rot(r_s{*}h, r_s{*}h)$, verify e_3, compute $r^{'}{}_s{*}h = r^{'}{*}h + rot(r^{'}{*}h, \lceil log_2(2q-1)\rceil' M)$,
get $id = (e_4 - r^{'}{}_s{*}h) \oplus m$	$\xleftarrow{e_4}$	$e_4 = r^{'}{}_s{*}h + id \oplus m$, $r{*}h \leftarrow r^{'}{*}h$

Fig. 1: The proposed protocol

to reader $e_4 = r^{'}{}_s{*}h + id \oplus m$, where $r^{'}{}_s{*}h = r^{'}{*}h + rot(r^{'}{*}h, \lceil log_2(2q-1)\rceil' M)$, and replaces $r{*}h$ with $r^{'}{*}h$. Note that, similarly to Equation (3) and from the description in section IV-A , in its polynomial form, e_4 can be written:

$$e_4 = (X^{HW(M)} + 1){*}r^{'}{*}h + m \oplus id \ mod(X^N - 1) \quad (4)$$

Let id_{temp} the binary polynomial defined by $id_{temp} = m \oplus id$, this means that $id_{temp_i} = m_i \oplus id_i$, where $0 \leq i \leq N-1$. The tag's next authentication session is executed with the new value $r^{'}{*}h$.
To counteract replay attacks, the reader retrieves the tag's identity by $id = (e_4 - r^{'}{}_s{*}h) \oplus m$ and not by the decryption of $e_4\ mod\ q$ using the private key $(g,\ f)$. Note that the reader/back-end can authenticate a tag without knowledge of the tag's identity. This feature is very much interesting in some applications, specially in supply chains. Off-line authentication of a tag (with no exchanges with a central database) is made possible as the same private key $(g,\ f)$ is used for every tag.

D. *Roaming support*

Inter-domain authentication of tags, especially in the supply chain environment is not well investigated in the literature. In [20], *Li et al.* propose a protocol based on hash function for the supply chain, but it was proven in [21] to be vulnerable to multiple-attacks. Our proposed protocol can be extended to securely support the inter-domain tag authentication. Just before moving to another domain, the last authentication to the domain leads to the tag being updated with the next domain public key $h^{'}$.

V. Security and privacy analysis

The resistance of our protocol against the classical security attacks is directly derived from the properties of NTRU and HMAC.
In the following analysis, we only consider vulnerabilities over the reader-tag channel. The channels between the reader and the back-end are considered as safe as both equipments have computing and battery resources, they are under the same administrative domain and they can implement any security protocols.

A. *Resistance to replay attaks*

For each authentication session, both the tag and reader/back-end generate new pseudo-random values m, $r_s{*}h$ and $r^{'}{*}h$, thus making messages randomized (personalized) for each session. A replay attack to the reader is unlikely to occur, as the reader/back-end is assumed implementing a good pseudo-random polynomial generator, so they are unlikely to generate the same $r^{'}{*}h$.
A replay attack to the tag can only be successful if the attacker prevents the tag from receiving $(e_2,\ e_3)$ (e.g. by transmitting a jamming signal) and the tag generates the same sequences m and $r_s{*}h$ in the next authentication attempt. The probability this attack is successful is negligible thanks to HMAC and the construction method of $r_s{*}h$, however, as discussed in the desynchronization attack, in case of success, it does not lead to any desynchronization between tag and reader/back-end.

B. *Resistance to man in the middle attacks*

We have demonstrated that, the attacker can not break NTRU using the modification that we have introduced on the choice of r (cf. section IV-A). For the attacker to authenticate as a legitimate tag or a legitimate reader. If the attacker is passive, the best is experimenting a replay attack. Indeed, suppose that the attacker wants to be authenticated as a legitimate tag, he has to produce a valid message e_4 or to retrieve the tag secret identity id. It's clear that the attacker cannot produce a valid e_4 from the previously exchanged messages between the tag and readers because any modification in a previous tag's response e_1 will be detected in the attacker's response e_4 thanks to the random value $r^{'}{*}h$ generated by the reader itself. On the other hand, the attacker cannot retrieve the identity of the tag because he cannot break NTRU.
If the attacker is active, he will try to take advantage of the fact that the number of $r_s{*}h$ between two legitimate authentication sessions is limited, to retrieve the identity of the tag id, he must first find the pseudo random value m or $r{*}h$ (initial values assigned to the tag during the previous mutual authentication session). For this, he queries the tag between two legitimate mutual authentication sessions. He has to retrieve the polynomial $m^{(k)} - m^{(i)}$ for a specific k and several i in order to find $m^{(k)}$ and $r{*}h$, where $m^{(j)}$ corresponds to the m of the j-th authentication attempt (attacker's query). Indeed, each $m^{(j)}$ is a binary

polynomial, so if the attacker finds a specific k and some $m^{(k)} - m^{(i)}$ for several i, he can deduce some coefficients of $m^{(k)}$ and he can proceeds by brute force attack to get the rest of coefficients of $m^{(k)}$, then he gets ${r_s}^{(k)}{*}h$, $m^{(k+1)}$, etc. and proceeds by another attack to deduce $r{*}h$ because from the Equation (3) it's clear that:

$$r{*}h = \frac{{e_1}^{(k)} - m^k}{X^i + X^j},\ (i,j) \in [\![0, \text{N-1}]\!]^2$$

Note that this attack is not be performed as he cannot retrieve for a specific k and several i, $m^{(k)} - m^{(i)}$. Indeed, suppose that the attacker begins his attack as described above, the tag responds to the i-th and to (i+1)-th attacker queries with ${e_1}^{(i)} = X^{k_i}{*}r{*}h + X^{k_{i-1}}{*}r{*}h + m^{(i)}\ mod(X^N - 1)$ and ${e_1}^{(i+1)} = X^{k_{i+1}}{*}r{*}h + X^{k_i}{*}r{*}h + m^{(i+1)}\ mod(X^N - 1)$, respectively (cf. Equation (3)), where $X^N{*}r{*}h\ mod(X^N - 1) = r{*}h$, and $(k_{i-1}, k_i, k_{i+1}) \in \mathbb{N}^3$. So,

$${e_1}^{(i+1)} - {e_1}^{(i)} = (X^{k_{i+1}} - X^{k_{i-1}}){*}r{*}h + m^{(i+1)} - m^{(i)}\ mod(X^N - 1) \quad (5)$$

Suppose that the attacker wants to obtain the value of $m^{(i)}$ from $e^{(i-1)}$, $e^{(i)}$, and $e^{(i+1)}$. Thanks to calculation with $mod(X^N - 1)$ and to simplify the reasoning we can write $k_i = \sum_{j=1}^{i} HW({r_t}^{(j)})$, then $k_{i+1} - k_{i-1} = (HW({r_t}^{(i+1)}) + HW({r_t}^{(i)}))$. To retrieve $m^{(i+1)} - m^{(i)}$ the attacker should eliminate the term $(X^{k_{i+1}} - X^{k_{i-1}}){*}r{*}h$ in the Equation (5), this is possible only if k_{i+1} equal to k_{i-1} or if each of k_{i+1} and k_{i-1} is a multiple of N. However, this is unlikely to occur as $k_{i+1} = k_{i-1}$ means that the tag has generated successively two null values of r_t but this scenario is unlikely to occur because HMAC generates good pseudo random sequences. On the other hand, the value of s is chosen in such a way that k_{i+1} and k_{i-1} are never a multiple of N in the same time, in other words $k_{i+1} - k_{i-1} = (HW({r_t}^{(i+1)}) + HW({r_t}^{(i)}))$ is less than N, so s can be chosen such that its length is less than $N/2$. Then the attacker cannot retrieve $m^{(i+1)} - m^{(i)}$ and $m^{(i)} - m^{(i-1)}$ to try the attack described above on $m^{(i)}$.
However, let's w the number of all possible values of an $r_s{*}h$ derived from one $r{*}h$ (initially assigned to the tag by the reader/back-end) between two legitimate authentication sessions, it's clear that $w = N + N - 1 + N - 2 + ... + N - (N - 1)$. Then, $w = N^2 - \sum_{i=1}^{N-1} i = 1/2(N^2 + N)$, for $N = 251$, $w = 31626$. Consequently, if the attacker queries the tag more than w times, there will certainly be ${e_1}^{(u_1)}$ and ${e_1}^{(v_1)}$ using the same $r_s{*}h$, so from the Equation (3) we deduce:

$${e_1}^{(u_1)} - {e_1}^{(v_1)} = m^{(u_1)} - m^{(v_1)}, (u_1, v_1) \in [\![1, \text{w}]\!]^2 \quad (6)$$

Note that, the attacker cannot accurately determine ${e_1}^{(u_1)}$ and ${e_1}^{(v_1)}$. If the attacker continues to query the tag (before the next mutual authentication), there will be ${e_1}^{(u_2)}$ and ${e_1}^{(v_2)}$ such that ${e_1}^{(u_2)} - {e_1}^{(v_2)} = m^{(u_2)} - m^{(v_2)}$. However, and thanks to HMAC it is likely that each of $m^{(u_1)}, m^{(u_2)}, m^{(v_1)}$ and $m^{(v_2)}$ is different from each other. If we continue this reasoning we note that the attacker is not able to find in reasonable period of time for a specific k, several i such that ${e_1}^{(k)} - {e_1}^{(i)} = m^{(k)} - m^{(i)}\ mod(X^N - 1)$. As such the attacker cannot retrieve either m or $r{*}h$.
The attacker cannot proceed by the previously described attack (cf. Equation (6)) on the message e_4 in order to retrieve $id \oplus m$ or $r^{'}{*}h$ because $r^{'}{*}h$ changes for each e_4. However, the tag next authentication session will be executed with $r^{'}{*}h$ which has been generated by the reader in the previous session (session k), so once the session k is successfully completed and before the next mutual authentication session happens, the attacker can start to query the tag. If he queries the tag more than $1/2(N + N^2)$ there *may be* ${e_1}^{(l)}$ such that ${e_1}^{(l)} = (X^{HW(M^{(k)})} + 1){*}r^{'}{*}h + m^{(l)}\ mod(X^N - 1)$ (cf. Equation (4)), then: ${e_1}^{(l)} - {e_4}^{(k)} = m^{(l)} - m^{(k)} \oplus id, (k,l) \in \mathbb{N}^2$ (cf. Equations (6)). Even if he finds the right ${e_1}^{(l)}$, it's clear that he cannot retrieve the identity of the tag id from this equation (he does not know either $m^{(k)}$ or $m^{(l)}$).
On the other hand, the identity of the tag is encrypted (cf. Equation (4)) in such a way that the multiple transmission attacks on NTRU is avoided. Indeed, if the same message m is transmitted many times using the same public key h, but with different random r's, the attacker will be able to retrieve a large part of the message NTRU [14]. To counteract this attack, we encrypt in e_4 $id \oplus m$ instead of the static id value, as $id \oplus m$ is different from one authentication session to another. As such, our protocol is resistant to man in the middle attacks.

C. Tag anonymity and resistance to tracking

Our protocol supports this challenging property for RFID systems. Two scenarios arise according to whether the attacker is passive or active. If the attacker is passive, i.e. he can not query the tag but he can eavesdrop communications between the tag and legitimate readers. He cannot track the tag as the messages exchanged over the wireless channel between the tag and the reader are randomized in each authentication session thanks to m, $r^{'}{*}h$ (cf. Equation (4)). If the attacker is active, i.e. he can query the tag. The privacy is also guaranteed as the tag respond $e_1 = r_s{*}h + m$ is randomized in each attacker's request thanks to the construction method of $r_s{*}h$ and HMAC. Indeed, the attacker cannot track the tag between two mutual authentication sessions by trying to distinguish the specific $r{*}h$ used by the tag because in ${e_1}^{(i)} = (X^{k_i} + X^{k_{i-1}}){*}r{*}h + m^{(i)}\ mod(X^N - 1)$, $(i,j) \in \mathbb{N}^2$ (cf. Equation (3)) the attacker cannot obtain information about specific coefficients of $r{*}h$ that allows him to track the tag.

D. Resistance to desynchronization attacks

The attacker can try to make the tag and reader/back-end out of synchronization by acting as an active man in the middle attacker, modifying e_2 in order to provide to the tag an invalid value of $r^{'}{*}h$, this modification will be necessarily detected in the HMAC value $e_3 = H_M(r^{'}{*}h)$ that guarantees the authenticity of $r^{'}{*}h$ because M know only to the tag and the reader which has the private key $(g,\ f)$ that allows decryption of e_1 (the cipher value of M

(or m)). Indeed, suppose that, the attacker has modified the bit at index k of e_2, with e_{20} being the least significant bit of e_2. He cannot produce a valid value of e_3 because he does not know the values of M (or m) and $r^{'} * h$ that must be used in this HMAC calculation. Then tag-reader/back-end synchronisation is guaranteed.

VI. Performance evaluation

Our proposed protocol is designed for low-cost RFID tags. All the complex operations such as polynomial multiplications, random polynomial generation in D_r, computation of modulo, etc. are done at the server.
Let $n = \lceil log_2(2q-1) \rceil N$, the tag only requires to store about $2N + n + 128$ bits and to support lightweight hash function like Photon [22] or Keccak [23]. The computations by the tag are limited to two HMAC values, and an average of about $\lceil log_2(2q-1) \rceil (N + S/2)$ bitewise right circular shift, addition of four polynomials, and $n + N$ bitwise xor operations. On the other hand, the reader/back-end implements right circular shift operator, and two NTRU algorithms to perform one encryption and one decryption each one with a complexity of only $O(N^2)$. Moreover, the reader/back-end does not require any exhaustive search in keys to authenticate a tag.

VII. Conclusions

In this paper, we propose an adaptation of NTRU for low-cost RFID tag, and a lightweight mutual authentication protocol based on this NTRU's adaptation. This solution satisfies the security and privacy requirements for RFID systems. It benefits from the NTRU features like high security level, and fast encryption and decryption. It provides remarkable properties such as strong scalability and untraceability, and resistance to known classical security attacks. Furthermore, it can be implemented efficiently into low-cost tags, as tags are only required to implement a lightweight hash function, and bit-wise operations in $GF(2)$.

References

[1] M. Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 66–75, January 1991. [Online]. Available: http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html

[2] S. Vaudenay, "On privacy models for rfid," in *ASIACRYPT*, 2007, pp. 68–87.

[3] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic Approach to "Privacy-Friendly" Tags," in *RFID Privacy Workshop*, MIT, Massachusetts, USA, November 2003.

[4] S. V. Kaya, E. Savaş, A. Levi, and Özgür Erçetin, "Public key cryptography based privacy preserving multi-context rfid infrastructure," *Ad Hoc Networks*, vol. 7, no. 1, pp. 136 – 152, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870508000036

[5] L. Batina, S. Seys, D. Singelee, and I. Verbauwhede, "Hierarchical ECC-Based RFID Authentication Protocol," in *RFIDSec'11*, Amherst, Massachusetts, USA, June.

[6] Y. K. Lee, L. Batina, D. Singelée, and I. Verbauwhede, "Low-Cost Untraceable Authentication Protocols for RFID," in *WiSec'10*, S. Wetzel, C. Nita-Rotaru, and F. Stajano, Eds., ACM. Hoboken, New Jersey, USA: ACM Press, March 2010, pp. 55–64.

[7] T. Sekino, Y. Cui, K. Kobara, and H. Imai, "Privacy enhanced rfid using quasi-dyadic fix domain shrinking," in *GLOBECOM*. IEEE, 2010, pp. 1–5.

[8] H. Niederreiter, "Knapsack-type cryptosystems and algebraic coding theory," *Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform.*, vol. 15, no. 2, pp. 159–166, 1986.

[9] M. Girault, "Low-size coupons for low-cost ic cards," in *CARDIS*, 2000, pp. 39–50.

[10] M. Girault, G. Poupard, and J. Stern, "On the fly authentication and signature schemes based on groups of unknown order," *J. Cryptology*, vol. 19, no. 4, pp. 463–487, 2006.

[11] M. Girault, L. Juniot, and M. Robshaw, "The Feasibility of On-the-Tag Public Key Cryptography," in *Workshop on RFID Security – RFIDSec'07*, Malaga, Spain, July 2007.

[12] A. Poschmann, M. J. B. Robshaw, F. Vater, and C. Paar, "Lightweight cryptography and rfid: Tackling the hidden overhead," *TIIS*, vol. 4, no. 2, pp. 98–116, 2010.

[13] M. McLoone and M. J. B. Robshaw, "New architectures for low-cost public key cryptography on rfid tags," in *ISCAS*, 2007, pp. 1827–1830.

[14] J. Hoffstein, J. Pipher, and J. H. Silverman, "Ntru: A ring-based public key cryptosystem," in *Lecture Notes in Computer Science*. Springer-Verlag, 1998, pp. 267–288.

[15] "Ieee standard specification for public key cryptographic techniques based on hard problems over lattices," *IEEE Std 1363.1-2008*, pp. C1 –69, 10 2009.

[16] "Lattice-based polynomial public key establishment algorithm for the financial services industry," *X9.98 standard*, 4 2011.

[17] A. Atici, L. Batina, J. Fan, I. Verbauwhede, and S. Yalcin, "Low-cost implementations of ntru for pervasive security," in *Application-Specific Systems, Architectures and Processors, 2008. ASAP 2008.*, july 2008, pp. 79 –84.

[18] "Consortium for efficient embedded security, efficient embedded security standard 1 version 2." [Online]. Available: http://grouper.ieee.org

[19] N. Howgrave-Graham, J. H. Silverman, and W. Whyte, "Choosing parameter sets for ntruencrypt with naep and sves-3," ser. CT-RSA'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 118–135.

[20] Y. Li and X. Ding, "Protecting rfid communications in supply chains," in *computer and communications security*, ser. ASIACCS '07, 2007, pp. 234–241.

[21] T. v. Deursen and S. Radomirovic, "Security of an RFID Protocol for Supply Chains," in *Proceedings of the 2008 IEEE International Conference on e-Business Engineering*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 568–573.

[22] J. Guo, T. Peyrin, and A. Poschmann, "The photon family of lightweight hash functions," in *CRYPTO*, 2011, pp. 222–239.

[23] E. B. Kavun and T. Yalcin, "A Lightweight Implementation of Keccak Hash Function for Radio-Frequency Identification Applications," in *RFIDSec'10*, ser. Lecture Notes in Computer Science, S. O. Yalcin, Ed., vol. 6370. Istanbul, Turkey: Springer, June 2010, pp. 258–269.