J. Math. Crypt. **3** (2009), 205–214 DOI 10.1515 / JMC.2009.011

# Algebraic attack on NTRU using Witt vectors and Gröbner bases

Gérald Bourgeois and Jean-Charles Faugère

Communicated by Jaime Gutierrez

**Abstract.** We present an algebraic attack on NTRU (restricted to the case where the parameter $q$ is a power of two) using the method of the Witt vectors proposed by Silverman, Smart and Vercauteren [17]; the latter considered only the first two bits of a Witt vector attached to the recovering of the secret key in order to reduce the problem to the resolution of an algebraic system over $\mathbb{F}_2$. The theoretical complexity of this resolution was not studied by the authors. In this paper, we use the first three bits of the Witt vectors to obtain supplementary equations which allow us to reduce the complexity of the attack. Using Gröbner basis complexity results of overdetermined systems, we have been able to provide a theoretical complexity analysis. Additionally we provide experimental results illustrating the efficiency of this approach. Moreover, we prove that the use of the fourth bit does not improve the complexity, what is surprising. Unfortunately, for standard values of the NTRU parameters, the proven complexity is around $2^{246}$ and this attack does not make it possible to find the private key.

**Keywords.** NTRU, algebraic attack, Gröbner bases, Witt vectors, FGb.

**AMS classification.** Primary 11T71, 13P10, 13K05.

## 1 Introduction

*Algebraic cryptanalysis* can be described as a general framework that permits to assess the security of a wide range of cryptographic schemes. The basic idea of such cryptanalysis is to model a cryptographic primitive by a set of polynomial equations. The system of equations is constructed such that the solution of this system is precisely the secret information of the cryptographic primitive (for instance, the secret key of an encryption scheme). It is a powerful technique that applies potentially to a wide range of cryptosystems; for instance, Faugère and Joux present, in [13], an effective algebraic cryptanalysis of the HFE cryptosystem by solving a system of 80 quadratic dense equations in 80 variables over $\mathbb{F}_2$.

The goal of this paper is to generalize the algebraic attack on NTRU by Silverman, Smart, Vercauteren, described in [17], and evaluate the complexity of the various associated attacks.

### 1.1 Basic facts about the NTRU cryptosystem

First, let us recall the NTRU problem over a ring field $\mathbb{Z}_q$, where $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ are the integers modulo $q$. The public parameters of NTRU (we refer to [15] for a complete description of NTRU) are: $N$ a prime number (for instance $N = 251$), $q = 2^m$ a power

of two (for instance $q = 128$). Consider the polynomial ring $\mathbb{R}_q = \mathbb{Z}_q[X]/(X^N - 1)$; $\star$ refers to the product in $\mathbb{R}_q$.

The key generation begins by choosing two random polynomials $F$ and $g$ in $\mathbb{F}_2[X]$ each of degree $N-1$. Note that the number of non zero coefficients of $F$ and $g$ can be fixed too, which provides some additional equations, but we do not take it into account here. The private key is the polynomial $f$ defined by $f = 1 + p \star F$, where $p = 2 + X$ and the public key is the polynomial $h$ defined by $h = p \star f_q^{-1} \star g$, where $f_q^{-1}$ is the inverse of $f$ in $\mathbb{R}_q$; this inverse almost surely exists because $F$ is randomly selected.

Hence, breaking the NTRU system is equivalent to the problem of recovering the polynomial $F \in \mathbb{F}_2[X]$ from a given polynomial $h$, knowing that $f \star h = (1 + p \star F) \star h = p \star g$.

**NTRU Problem**
**Input:** positive integers $N, q = 2^m$, and a polynomial $h$ of degree less than $N-1$ in $\mathbb{Z}_q[X]$.
**Problem:** find if there is a pair $(F, g) \in \mathbb{F}_2[X]^2$ such that

$$\deg(F) < N \text{ and } \deg(g) < N,$$
$$(1 + p\,F)\,h \equiv p\,g \mod \left(X^N - 1\right) \mod 2^m, \tag{1}$$

where $p = 2 + X$.

### 1.2 Idea of the algebraic attack

It is easy to see that the NTRU problem can be modeled as an MQ problem, that is the problem of finding one (not necessarily all) solution to a system of $m$ multivariate quadratic equations with $n$ variables over $\mathbb{F}_q$: Assume that $g(X) = \sum_{i=0}^{N-1} g_i X^i$ and $F(X) = \sum_{i=0}^{N-1} F_i X^i$, where $F_i, g_i \in \mathbb{F}_2$ are unknown. From equation (1) we consider the polynomial

$$Z(X) = (1 + (2 + X)\,F(X))\,h - (2 + X)\,g(X).$$

The coefficients of the remainder of $Z(X)$ w.r.t. $X^N - 1$ lead to $N$ *linear* equations

$$\text{Rem}(Z(X), X^N - 1) = \sum_{i=0}^{N-1} e_i\,X^i,$$
$$e_1 = \ldots = e_{N-1} = 0 \mod 2^m. \tag{2}$$

Of course, since the equations (2) are true modulo $2^m$ they are also true modulo 2, 4, 8, 16. The first idea is to consider these equations modulo 2 since we can compute directly a Gröbner basis over $\mathbb{F}_2$. However, this method is rather naive since we obtain a system of $2\,N$ quadratic equations in $N$ variables (since we add the field equations $x^2 - x = 0$). Moreover, in that case, the corresponding algebraic system seems to behave like a random algebraic system. Recall that the resolution of a system of $N$

quadratic equations in $N$ variables in $\mathbb{F}_2$ is an NP-hard problem (see [14]) and that it was experimentally established that the resolution of a random system is a hard problem. Even if the equations that we get have symmetries (they are invariants by the cyclic group for instance), it is an open issue to obtain efficient Gröbner basis algorithms dealing with symmetries. Consequently, it is very unlikely to obtain an efficient attack in this way.

The next idea is to extract *more information* from equations (2): In a first step we consider these $e_i = 0 \mod 2^{m'}$ with $m' < m$ (in practice $m' = 2, 3$ or 4), then we *transport* these equalities in the truncated Witt ring (for the definition of the Witt ring see Section 2.1). The problem is then reduced to an algebraic system with $N$ variables in $\mathbb{F}_2$. We hope to obtain an algebraic system that is easier to solve since we add additional equations and thus we obtain an overdetermined algebraic system (under some regularity assumption, Gröbner basis algorithms are much more efficient when the number of equations is much bigger than the number of variables). The main drawback of this method is that for a fixed $m'$ we add non-sparse polynomial equations of degree $2^{m'-1}$.

### 1.3 The results

In [17] the authors choose $m' = 2$ (that is they use only the first two bits of the total Witt vectors). The recovering of the secret key is reduced to solve a system of $N$ quadratic equations in $N$ variables in $\mathbb{F}_2$. In this paper, we use the third and fourth bit of the Witt vectors ($m' = 3$ and 4); we thus obtain $N$ additional equations of degree four for the third bit and $N$ additional equations of degree eight for the fourth bit. These choices provide systems of $2N$ or $3N$ equations in $N$ variables in $\mathbb{F}_2$.

In [17] the theoretical complexity of this problem is not studied. Here we carry out this computation and we prove that for $N = 251$ the use of the third bit divides the computing time by sixty, but surprisingly the use of the fourth bit gives no further improvement. However, these complexities are very large and this attack is unfeasible in practice.

We have conducted experiments with $20 \leq N \leq 25$ using the FGb/Maple software (similar results can be checked using the Magma computer algebra system): if we consider only the $2N$ first equations ($m' = 3$), then compared to the system reduced to $N$ first equations, the computing times are divided by two for $N = 25$ and the gap widens when $N$ increases.

When $m' = 4$, the additional equations of degree 8 are taken into account during the Gröbner basis process only starting from $N \geq 41$. We cannot test this contribution because of a lack of memory.

## 2 Recovering the secret key reduces to an algebraic system

### 2.1 The ring of Witt vectors

With the 2-adic numbers we can introduce the Witt vectors [9], but here we will only use $W_4[\mathbb{F}_2]$, the ring of Witt vectors over $\mathbb{F}_2$ of length 4; it is the set ${\mathbb{F}_2}^4$ provided with

a ring structure, the bijection onto the ring $\mathbb{Z}_{2^4}$,

$$\left(\begin{array}{rcl} W_4[\mathbb{F}_2] & \longrightarrow & \mathbb{Z}_{2^4} \\ a=[a_0,a_1,a_2,a_3] & \longmapsto & \sum_{i=0}^{3} a_i 2^i \end{array}\right),$$

induces ring operations $+$ and $\star$ on the set $W_4[\mathbb{F}_2]$ (see [5]).

Note that if we consider the Witt ring with coefficients in $\mathbb{F}_p, p \geq 3$, then the Witt isomorphism is much more complicated (see Teichmüller representatives in [16]).

### 2.2 First part of the construction of the algebraic equations

We rewrite the equality $f \star h = p \star g \mod 2^4$ in $W_4(\mathbb{F}_2)$ with the following notations:

$$g = \sum_{i=0}^{N-1} g_i X^i, \quad \text{where } g_i \in \mathbb{F}_2 \text{ is unknown (we use the notation } g_{-1} = g_{N-1}),$$

$$F = \sum_{i=0}^{N-1} F_i X^i, \quad \text{where } F_i \in \mathbb{F}_2 \text{ is unknown,}$$

$$h = \sum_{i=0}^{N-1} h_i X^i, \quad \text{where } h_i = [h_{i0}, h_{i1}, h_{i2}, h_{i3}] \in W_4(\mathbb{F}_2) \text{ is given.}$$

The unknowns are the $(g_i)$ and $(F_i)$ but we only need $(F_i)$.

It follows that

$$f = (1 + (2 + X)\, F) = (1 + 2F_0 + F_{N-1}) + \sum_{i=1}^{N-1} (2F_i + F_{i-1}) X^i = \sum_{i=0}^{N-1} f_i X^i,$$

where $f_i \in W_4(\mathbb{F}_2)$ is defined by

$$\left\{\begin{array}{rcl} f_0 & = & [1 + F_{N-1}, F_0 + F_{N-1}, F_0\, F_{N-1}, 0], \\ f_i & = & [F_{i-1}, F_i, 0, 0] \text{ if } i \geq 1. \end{array}\right.$$

In the same way,

$$p \star g = (2 + X) \star g = \sum_{i=0}^{N-1} (2g_i + g_{i-1}) X^i = \sum_{i=0}^{N-1} R_i X^i,$$

where $R_i = [g_{i-1}, g_i, 0, 0] \in W_4(\mathbb{F}_2)$.

Finally,

$$f \star h = \sum_{k=0}^{N-1} L_k X^k \quad \text{with} \quad L_k = \sum_{(i+j=k) \bmod N} f_i h_j.$$

Then for all $k \leq N-1$, $L_k = R_k$, that is, $[L_{k0}, L_{k1}, L_{k2}, L_{k3}] = [g_{k-1}, g_k, 0, 0]$. The equations $L_{k0} = g_{k-1}, L_{k1} = g_k$ were used in [17]; in our work we add the equalities $L_{k2} = 0, L_{k3} = 0$.

### 2.3 Expressions of $S_0, S_1, S_2, S_3$ in the case of a sum of more than two terms

The expression $L_k$ is the sum of $N$ terms, each one being the product of two terms. To compute $L_k$, we must therefore determine the product $P(a,b)$ of two elements $a,b$ of the Witt's ring as well as the sum $S(a_1,\ldots,a_s)$ of $s$ elements $a_1,\ldots,a_s$ of the Witt's ring. Straightforward computations involve the following formulas.

$$\begin{aligned}
P_0(a,b) &= a_0b_0,\\
P_1(a,b) &= a_0b_1 + a_1b_0,\\
P_2(a,b) &= a_0b_0a_1b_1 + a_0b_2 + b_0a_2 + a_1b_1,\\
P_3(a,b) &= a_0b_0a_1b_1a_2 + a_0b_0a_1b_1b_2 + a_0b_0a_1b_1 + a_0b_0a_2b_2 + a_0a_1b_1b_2 + a_0b_3\\
&\quad + b_0a_1b_1a_2 + b_0a_3 + a_1b_2 + b_1a_2.
\end{aligned}$$

The computations for $S$ are more intricate.

**Definition** (See [5]). Let $a_1,\ldots,a_s$ be elements of the Witt's ring. We define the *weight* of a monomial in the $(a_{ik})_{ik}$ as follows: *weight*$(a_{ik}) = 2^k$ for all $i,k$ and the *weight* of a product of two monomials is the sum of their *weight*s.

**Theorem 2.1** (See [5]). *$S_r(a_1,\ldots,a_s)$ is the sum of all monomials of weight $2^r$ in the $(a_{ik})_{ik}$.*

Now we can deduce the four components of $S$.

$$S_0(a_1,\ldots,a_s) = \sum_i a_{i0}, \tag{3}$$

$$S_1(a_1,\ldots,a_s) = \sum_{i<j} a_{i0}a_{j0} + \sum_i a_{i1}, \tag{4}$$

$$\begin{aligned}
S_2(a_1,\ldots,a_s) &= \sum_i a_{i1}\sum_{i<j} a_{i0}a_{j0} + \sum_{i<j} a_{i1}a_{j1} + \sum_i a_{i2}\\
&\quad + \sum_{i<j<k<l} a_{i0}a_{j0}a_{k0}a_{l0},
\end{aligned} \tag{5}$$

$$\begin{aligned}
S_3(a_1,\ldots,a_s) &= \sum_i a_{i3} + \sum_{i<j} a_{i2}a_{j2} + \sum_i a_{i2}\sum_{i<j} a_{i1}a_{j1} + \sum_{i<j<k<l} a_{i1}a_{j1}a_{k1}a_{l1}\\
&\quad + \sum_{i<j} a_{i0}a_{j0}\sum_i a_{i1}\sum_i a_{i2} + \sum_{i<j} a_{i0}a_{j0}\sum_{i<j<k} a_{i1}a_{j1}\,a_{k1}\\
&\quad + \sum_{i<j<k<l} a_{i0}a_{j0}a_{k0}a_{l0}\left(\sum_i a_{i2} + \sum_{i<j} a_{i1}a_{j1}\right)\\
&\quad + \sum_i a_{i1}\sum_{i_1<i_2<i_3<i_4<i_5<i_6} a_{i_1 0}a_{i_2 0}a_{i_3 0}a_{i_4 0}a_{i_5 0}a_{i_6 0}\\
&\quad + \sum_{i_1<i_2<i_3<i_4<i_5<i_6<i_7<i_8} a_{i_1 0}a_{i_2 0}a_{i_3 0}a_{i_4 0}a_{i_5 0}a_{i_6 0}a_{i_7 0}a_{i_8 0}.
\end{aligned} \tag{6}$$

### 2.4 The $N$ equations associated to the third bit

In the following sums, the indices $u, u^*$ satisfy the relation $(u + u^* = k) \mod N$.

According to the formulas (3), (4), (5) giving $S_0, S_1, S_2$, the relations $L_{k2} = 0$ are written in $W_4[\mathbb{F}_2]$ in the form

$$\sum_i (f_{i0}h_{i^*1} + f_{i1}h_{i^*0}) \sum_{i<s} f_{i0}h_{i^*0}f_{s0}h_{s^*0} + \sum_{i<s} (f_{i0}h_{i^*1} + f_{i1}h_{i^*0})(f_{s0}h_{s^*1} + f_{s1}h_{s^*0})$$
$$+ \sum_i (f_{i0}h_{i^*0}f_{i1}h_{i^*1} + f_{i0}h_{i^*2} + f_{i1}h_{i^*1} + f_{i2}h_{i^*0})$$
$$+ \sum_{i<j<s<t} f_{i0}h_{i^*0}f_{j0}h_{j^*0}f_{s0}h_{s^*0}f_{t0}h_{t^*0} = 0.$$

In [17] the relations $L_k = R_k$ for the first two bits have been explicitly written as functions of the $(F_i)$, which allows to highlight a certain symmetry in the equations which are quadratic. However, this property does not seem to speed up the computation. Here we carry out these computations, these $N$ equations have total degree four in the $(F_i)$. To give an idea of the complexity, if $N = 17$ for instance, according to tests carried out, each equation contains *hundreds of terms*.

### 2.5 The $N$ equations associated to the fourth bit

According to the formulas (3), (4), (5), (6) giving $S_0, S_1, S_2, S_3$, the relations $L_{k3} = 0$ can be written in $W_4[\mathbb{F}_2]$, but they are much more intricate. These can be also obtained by implementing general Witt vector arithmetic.

Here we obtain $N$ equations of total degree eight in the $(F_i)$. Once again, in the case $N = 17$, and according to the tests carried out, each equation contains *thousands of terms*.

## 3 Analysis of the theoretical complexity

In [17], the computations and complexity of the NTRU algebraic attack was evaluated using relinearization (XL [8], or one of its variants XLS and FXL [8]). Here, we propose to use a more efficient tool for solving algebraic systems, namely fast Gröbner bases [7, 6] algorithms: $F_4$ [10] (also available in the Magma computer algebra system) or the most recent $F_5$ algorithm [11]. In particular, it has been proved [1] – from both a theoretical and practical point of view – that XL [8] is less efficient than $F_5$. Due to the range of examples that become computable with $F_5$, Gröbner basis can be considered as a reasonable computable object in real scale applications. For several systems arising in cryptography, $F_5$ has achieved good results (for instance on HFE [13]).

Now, we consider the concept of a semi-regular sequence of polynomials; approximately, it is a generic sequence of polynomials. The mathematical definition is given in [4, p. 9] and we use the following three properties.

i) In [4, p. 10] and [3] the authors conjecture, in agreement with numerical tests, that if we randomly select a system containing equations in a large number of variables in $\mathbb{F}_2$, then the associated sequence is almost surely semi-regular.

ii) If we want to solve (using the $F_5$ algorithm [11]) a system in $N$ variables which is a semi-regular sequence, then a theoretical estimation of the complexity of this problem is computed in [4, p. 17] and [3]: Let $d_{\text{reg}}$ be the index of regularity and $\binom{N}{m}$ be the number of $m$-combinations of $\{1,\ldots,N\}$, then, using sparse linear algebra techniques, the complexity of solving the associated linear system is

$$O\left(\binom{N}{d_{\text{reg}}}^2\right). \tag{7}$$

iii) Moreover, the value of $d_{\text{reg}}$ is given explicitly, in [4, p. 12] and [3], as a function of the number of equations and their degrees:

**Theorem 3.1.** *For a semi-regular system $(f_1,\ldots,f_m)$ such that $d_i = \deg(f_i)$, the index of regularity $d_{\text{reg}}$ is the first non-negative coefficient in the power series*

$$H(z) = \prod_{i=1}^{m} \left(\frac{1}{1+z^{d_i}}\right) (1+z)^n .$$

We return to our systems in $N$ variables, the tests carried out for $N \leq 25$ are in agreement with the results obtained by applying the calculations exposed in ii) and iii). Thus, in the following we assume that, during our attack, the considered systems are semi-regular sequences.

- Case 1: two bits are used ($m' = 2$). In that case $m = n = N$ and $d_i = 2$ so that

$$H(z) = \left(\frac{1+z}{1+z^2}\right)^N .$$

- Case 2: three bits are used ($m' = 3$). In that case $n = N$ , $m = 2N$ and $d_i = 2$ (resp. $d_i = 3$) when $i \leq N$ (resp. $i > N$) and we have

$$H(z) = \left(\frac{1+z}{(1+z^2)(1+z^4)}\right)^N .$$

- Case 3: four bits are used ($m' = 4$). In that case $n = N$ , $m = 3N$ and

$$d_i = \begin{cases} 2 & \text{if} \quad 1 \leq i \leq N, \\ 3 & \text{if} \quad N+1 \leq i \leq 2N, \\ 4 & \text{if} \quad 2N+1 \leq i \leq 3N, \end{cases}$$

and we have

$$H(z) = \left(\frac{1+z}{(1+z^2)(1+z^4)(1+z^8)}\right)^N .$$

In the following tabular, we report the value of $d_{\text{reg}}$ for the usual NTRU parameters:

| $d_{\text{reg}} \backslash N$ | 2 bits | 3bits | 4bits |
|---|---|---|---|
| 251 | 29 | 28 | 28 |
| 503 | 53 | 52 | 52 |

From formula (7) it is easy to estimate the computational complexity $\tau$ for the same parameters:

| $\tau \backslash N$ | 2 bits | 3bits | 4bits |
|---|---|---|---|
| 251 | $2^{252}$ | $2^{246}$ | $2^{246}$ |
| 503 | $2^{480.5}$ | $2^{474}$ | $2^{474}$ |

When $N = 251$ (standard security parameter for NTRU) and when we use three bits instead of two bits, then the complexity is theoretically divided by sixty. On the other hand, considering the fourth bit is (almost) useless; this last result can seem paradoxical but it comes from the introduction of equations of very high degree.

**Remark.** According to [2, p. 19], the cryptanalysis of the HFE challenge in [13] was effective because the associated sequence was *not semi-regular* and then was much easier to solve.

## 4 Experiments with two and three bits

The performance of computations was measured by using a 1.6 GHz Intel Pentium IV processor provided with 1 GB RAM. During the practical tests it is possible to work with FGb inside the general computer algebra system Maple (Maple 12 see [12]), however, some of the Gröbner bases computations which follow were carried out by the second author using the $\text{F}_5$ algorithm. For $N = 25$, $q = 128$, we choose randomly $F$ and $g$ of degree 24 without fixing the number of non-zero terms (all random examples provide similar results). The quadratic system obtained in [17], thanks to the first two bits, admits between one and five solutions and is solved in 15 sec, which is almost the computing time of a random system of the same size. Of course, it is easy to recover the solution which leads to the private key. If we take the third bit into account, the 25 additional equations ($L_{k2} = 0$) are developed according to $(F_i)$ in few minutes on a PC using Maple. Now, we have to solve a system of 50 equations with 25 variables, there is only one solution, obtained in 8 sec. For $N = 25$ the computing time is divided by two. Moreover, this last factor increases with $N$ (experiments done for $N \leq 26$). Here the maximal degree observed during the computation is five (in accordance with the theoretical bound given by theorem 3.1) and the introduction of the fourth bit is useless because the $N$ supplementary equations induced by $L_{k3} = 0$ are of degree eight. If $N = 27$ then the maximal degree six is reached but the experiment fails because of a lack of memory. Thus we cannot test the contribution of the fourth bit.

## 5 Conclusion

We have completely tested the attack on NTRU using Witt vectors and Gröbner bases. In [17] only the first two bits of the Witt vectors were considered. Here, using the first three bits, we obtain $N$ additional equations of degree four, so the system becomes overdetermined but unfortunately the associated sequence behaves like a semi-regular sequence of polynomials. For standard values of the parameter, such as $N = 251$, we have shown that the computing time is divided by sixty. Yet, complexity for solving this system is around $2^{246}$ and we are unable to break NTRU.

If we consider the fourth bit, one observes that adding equations of degree eight does not decrease the complexity of the attack.

Thus, the algebraic attack using Witt vectors is not effective if one wants to solve the associated system using Gröbner basis algorithms.

## References

[1] Gwénolé Ars, Jean-Charles Faugère, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita, *Comparison between XL and Gröbner basis algorithms*. ASIACRYPT 2004 (Pil Joong Lee, ed.), LNCS 3329, pp. 338–353. Springer, December 2004.

[2] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy, *Asymptotic expansion of the degree of regularity for semi-regular systems of equations*. Mega 2005 (P. Gianni, ed.), Sardinia (Italy), 2005.

[3] ———, *On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations*. Proc. International Conference on Polynomial System Solving (ICPSS), pp. 71–75, 2004.

[4] ———, *Complexity of Gröbner basis computation for semi-regular overdetermined sequences over $\mathbb{F}_2$ with solutions in $\mathbb{F}_2$*. Technical report, INRIA research report, 2003.

[5] N. Bourbaki, *Algèbre Commutative*, Ch. IX., Masson 1983 or Springer 2006.

[6] B. Buchberger. *Gröbner Bases: an Algorithmic Method in Polynomial Ideal Theory. Recent trends in multidimensional systems theory*. Reider ed. Bose, 1985.

[7] B. Buchberger, G.-E. Collins, and R. Loos, *Computer Algebra Symbolic and Algebraic Computation*. Springer-Verlag, 2nd edition, 1982.

[8] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir, *Efficient algorithms for solving overdefined systems of multivariate polynomial equations*. Advances in Cryptology – EUROCRYPT 2000, LNCS 1807, pp. 392–407. Springer, 2000.

[9] I.V. Dolgachev, *Witt vector*, in: Hazewinkel, Michiel, Encyclopaedia of Mathematics, Kluwer Academic Publishers, 2001, ISBN: 978-1556080104.

[10] Jean-Charles Faugère, *A new efficient algorithm for computing Gröbner bases (F4)*, Journal of Pure and Applied Algebra 139 (1999), pp. 61–88.

[11] ———, *A new efficient algorithm for computing Gröbner bases without reduction to zero (F5)*. Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation ISSAC (T. Mora, ed.), pp. 75–83. ACM Press, July 2002, ISBN: 1-58113-484-3.

[12] ———, INRIA, Paris 6. FGb software downloadable at `http://www.grobner.org/jcf/Software/FGb/index.html`

[13] Jean-Charles Faugère and Antoine Joux, *Algebraic cryptanalysis of Hidden Field Equation (HFE) cryptosystems using Gröbner bases*. Advances in Cryptology – CRYPTO 2003 (Dan Boneh, ed.), LNCS 2729, pp. 44–60. Springer, 2003.

[14] Michael R. Garey and David S. Johnson, *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, 1979.

[15] J. Hoffstein, J. Pipher, and J. H. Silverman, *NTRU: a ring-based public key cryptosystem*. Algorithmic number theory – ANTS III, LNCS 1423, pp. 267–288. Springer, 1998.

[16] J. P. Serre, *Local Fields*, Graduate Texts in Mathematics 67, Sec. II.6. Springer-Verlag, Berlin, New York, 1979.

[17] J. H. Silverman, N. P. Smart, and F. Vercauteren, *An algebraic approach to NTRU ($q = 2^n$) via Witt vectors and overdetermined systems of non linear equations*. Security in Communication Networks – SCN 2004, LNCS 3352, pp. 278–298. Springer, January 2005.

Received 30 December, 2008; revised 27 October, 2009

**Author information**

Gérald Bourgeois, Département de Mathématiques,
Université de la Polynésie Francaise, BP 6570
98702 Faa'a, Tahiti, French Polynesia, France.
Email: bourgeois.gerald@gmail.com

Jean-Charles Faugère,
INRIA, Centre Paris-Rocquencourt, SALSA Project
UPMC, Univ Paris 06, LIP6
CNRS, UMR 7606, LIP6
Université Pierre et Marie Curie Paris 6
UFR Ingénierie 919
LIP6 Passy Kennedy, bureau 733
Boite Courrier 169
4, Place Jussieu 75252 Paris cedex 05, France.
Email: Jean-Charles.Faugere@inria.fr