Gröbner Bases for Public Key Cryptography

Massimo Caboara, Fabrizio Caruso, Carlo Traverso
Dipartimento di Matematica, Università di Pisa
{caboara,caruso,traverso}@dm.unipi.it

ABSTRACT
Up to now, any attempt to use Gröbner bases in the design of public key cryptosystems has failed, as anticipated by a classical paper of B. Barkee et al.; we show why, and show that the only residual hope is to use binomial ideals, i.e. lattices.

We propose two lattice-based cryptosystems that will show the usefulness of multivariate polynomial algebra and Gröbner bases in the construction of public key cryptosystems. The first one tries to revive two cryptosystems Polly Cracker and GGH, that have been considered broken, through a hybrid; the second one improves a cryptosystem (NTRU) that only has heuristic and challenged evidence of security, providing evidence that the extension cannot be broken with some of the standard lattice tools that can be used to break some reduced form of NTRU.

Because of the bounds on length, we only sketch the construction of these two cryptosystems, and leave many details of the construction of private and public keys, of the proofs and of the security considerations to forthcoming technical papers.

Categories and Subject Descriptors
L1.2 [Algorithms]: Algebraic Algorithms; E.3 [Data Encryption]: Public key cryptosystems

General Terms
Algorithms

1. BARKEE WAS RIGHT: INTRODUCTION
In 1994 an article appeared in the Journal of Symbolic Computation [2], with a remarkable title, and an even more remarkable set of authors:

*Supported by MURST under the PRIN project “Agebra commutativa, computazionale e combinatorica”. The research has originated from the activities of the “Special semester on Gröbner bases”, held in Linz at RICAM in 2006.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.
Copyright 2008 ACM 978-1-59593-904-3/08/07 ...$5.00.
Boo Barkee, Deh Cac Can, Julia Ecks, Theo Moriarty, R. F. Ree: “Why you cannot even hope to use Gröbner Bases in Public Key Cryptography: an open letter to a scientist who failed and a challenge to those who have not yet failed” (partially supported by Spectre)

The title of the paper is a bit misleading; it does not discuss the undeniable possibilities of Gröbner bases in cryptanalysis, but only the use of Gröbner bases in the construction of a public key cryptosystem. More precisely, it just shows how any attempt to exploit in a public key cryptosystem the worst-case complexity of Gröbner bases is deemed to fail, since computing a Gröbner basis and finding a normal form are quite different problems; decyphering cannot be more complex than encyphering, in a dense complexity model, hence the worst case complexity cannot be used to prevent attacks to the cryptogram based on linear algebra and partial Gröbner basis computation.

The cryptosystem considered (or rather a cryptosystem scheme) is the following:
1. The private key consists in a (multivariate) polynomial ring $R = k[X]$, a term-ordering, and an ideal $I$ given through a Gröbner basis $G$ with respect to the term-ordering, and a set $M$ of power products of $R$ that are in normal form mod $I$.
2. The public key consists in $R$, $M$ and a finite subset $F \subseteq I$. The term-ordering is not public.
3. Messages are linear combinations with coefficients in $k$ of elements of $M$.
4. A cryptogram $c$ is obtained adding to a message $m$ an element of the ideal generated by $F$, i.e. $c = m + \sum \phi_i f_i$, $\phi_i \in R$, $f_i \in F$.
5. Decyphering is done computing the normal form $c \Rightarrow_G m$.

Cryptosystems of this type should be called Barkee cryptosystems, but are rather known as Polly Cracker [6], that originally identified a special subclass independently defined. These are the concluding remarks of Barkee:

“The high complexity of Gröbner bases is in fact strictly related with the existence of polynomials in an ideal whose minimal degree representation in terms of a given basis is doubly exponential in the degree of the basis elements. Since such polynomials cannot be used as encoded messages, a cryptographic scheme applying the complexity of Gröbner basis to an ideal membership problem is bound to fail.

Is our reader able to find a scheme which overcomes this difficulty?
“In particular our reader could think (perhaps with some reason) that a sparse scheme could work. We believe (perhaps without reason) that sparsity will make the scheme easier to crack. We would be glad to test our belief on specific sparse schemes.”

It turned out that many other scientists have failed in the subsequent years. A recent survey of Franziska Löw has shown that the extreme sparsity is the common weakness of all the Polly Cracker cryptosystems. We detail here the argument, that is only sketched in [13], and generalizes methods of [11]. The argument is also reminiscent of the “symbolic reduction” phase of the F4 algorithm [5].

Remark that in the computation of the cryptogram \( c = m + \sum \phi_i f_i \), representing polynomials as sum of monomials and applying the distributive law, a finite set \( S \) (of moderate size) of monomials appears. If we can bound this support of the computation, we can reconstruct for each \( \phi_i \) a set of monomials containing their support. The \( \phi_i \) can then be reconstructed by linear algebra. Our aim is hence to find a small set \( S' \supseteq S \).

We assume first that every polynomial of \( F \), as well as any other polynomial “easily” obtained from \( F \) has at least three monomials.

Because of the sparsity, i.e. the extremely low probability that a monomial appears in the support of a polynomial, whenever we add a monomial multiple \( \mu = x^\alpha f_i \) of \( f_i \) to an already constructed polynomial \( \beta \), it is extremely unlikely that more than one monomial of \( f \) and one of \( \mu \) cancel; the exceptions should be part of the design, and in this case the design itself has to be public, hence it is likely to introduce new types of attacks.

As a result, the new polynomial contains more monomials than \( f \). This restricts the number of monomials in \( S \) and enlarge it until we can reconstruct the \( \phi_i \).

In the cryptogram \( c \), and hence in \( S' \), there will be pairs of monomials that also appear in some \( x^\alpha f_i \); we add to \( S' \) the support of \( x^\alpha f_i \), and we continue until we succeed in deciphering \( c \) via linear algebra, or until the procedure fails to enlarge \( S' \) (or the resources are exhausted).

This reasoning breaks if \( F \) contains monomials or binomials. In this case, a reduction may replace one monomial with another, or even with nothing; deciphering, one can perform long chains of reductions without exponential growth. An eavesdropper can only assume that every operation has one element in common with the others, but this does not help: every reconstruction has multiple possibilities, and the possibility of long simplification chains entail an exponential reconstruction. But another idea is possible.

If \( F \) contains monomials or binomials, or if monomials or binomials can be easily obtained from \( F \), and moreover the Gröbner basis of the ideal \( J \) generated by these “easy” monomials and binomials can be computed, instead of using \( k[X] \) and its monomials, we can use \( k[X]/J \) and its standard monomials, and see everything inside it. The attack is hence easier.

The only residual possibility is therefore that \( F \) itself is composed of monomials and binomials, i.e. it is a 2-nomial ideal (a 2-nomial is \( aX^n + bX^m \), and the Gröbner basis is difficult to compute.

Since moreover the constants \( a \) and \( b \) can be seen as additional variables, we can restrict the discussion to binomials, i.e. polynomials being the difference of two power products \( X^a - X^b \), and we can even assume that in \( k[X]/J \) the variables are invertible, since this can be handled e.g. adding one variable \( T \) and one polynomial \( 1 \cdot T \prod X_i \).

These assumptions allow to use alternative ideal representations that do not restrict seriously the class of examples.

2. LATTICE POLLY CRACKER

2.1 Binomial ideals and lattices

Let \( I \) be a binomial ideal. We can associate with each binomial \( b = X^a - X^b \in I \) the vector \( v_b = \beta - \beta \in \mathbb{Z}^n \). Conversely, given an element \( v \in \mathbb{Z}^n \) define \( \alpha_v = \max(0, v_i) \), \( \beta_v = -\min(0, v_i) \), and the binomial \( b_v = X^{\alpha_v} - X^{\beta_v} \). Clearly, \( v b_v = v \), while \( b = b_v b' \), where \( b' \) is the GCD of the two monomials of \( b \).

From a binomial ideal \( I \) we obtain a lattice, and from a lattice we obtain a binomial ideal; going from a binomial ideal to the corresponding lattice and back, we obtain the saturation \( I : \prod X_i = k[X, X^{-1}] / I[f(X)] \), that is a binomial ideal. We call a binomial ideal saturated with respect to the variables a Lattice Ideal.

A lattice usually is defined as a free Abelian group with a scalar product. We are rather interested in full-rank integer lattices, i.e. sublattices of \( \mathbb{Z}^n \) of rank \( n \), and the scalar product is not used explicitly, hence may be changed depending of the problem. The changes of coordinates in \( \mathbb{Z}^n \) have to be unimodular, but not necessarily orthogonal.

Given a set of generators of an integer lattice, i.e. a set of elements of \( \mathbb{Z}^n \), we have a corresponding ideal generated by the binomials corresponding to the vectors. This ideal might not be variable-saturated, but if the generating set contains a vector \( \alpha \) such that all its components are positive, then the ideal contains \( X^{\alpha} - 1 \), and this shows that every variable is invertible in the quotient, hence the ideal is variable-saturated, and corresponds to the ideal associated to the lattice.

A term ordering defines a total ordering on the lattice, and a set of elements of the lattice is a Gröbner basis if the corresponding set of binomials is a Gröbner basis.

A Polly Cracker system based on a lattice ideal has some peculiarities. Since the normal form of a monomial modulo a binomial ideal is a monomial with the same coefficient, enciphering a polynomial is equivalent to enciphering its monomials (no interaction can happen between the monomials).

Because of this, a Polly Cracker based on a lattice ideal can be seen without any reference to polynomials and ideals, hence we will speak of “Lattice Polly Cracker”.

2.2 Quotient of lattices and normal forms

Given a lattice \( L \subset \mathbb{Z}^n \) of rank \( n \) one wants to represent the elements of \( \mathbb{Z}^n / L \) in a canonical form. There are three families of such normal forms:

Minkowski normal form with respect to a basis \( B = \{b_i\} \) of the lattice. For a vector \( v \in \mathbb{Z}^n \), with \( v = \sum a_i b_i \), the normal form is given by \( \mu_B(v) = v - \sum a_i b_i \), where \( a_i = \lfloor a_i \rfloor \), i.e. \( a_i \) rounded to the nearest integer. Instead of rounding, floor or ceiling can be used, giving variants of the normal form.
**Voronoi normal form** with respect to a metric. It associates \( v \) to the vector \( \beta(v) = v - v' \), where \( v' \) is the lattice vector closest to \( v \) with respect to the metric (a sparring method has to be designed when the closest vector is not unique).

**Gröbner normal form** with respect to a product-compatible total ordering \( \tau \) on \( \mathbb{Z}^n \). The ordering has to have 0 as minimum for \( \mathbb{N}^n \) and is usually computed through a Gröbner basis \( G \) of \( L \). It is denoted by \( \gamma_G(v) \) or \( \gamma(v) \), or simply \( \tau(v) \), and is the \( \tau \)-smallest positive element equivalent to \( v \).

It is sometimes useful to consider shifted Gröbner normal forms, \( \gamma_{G,\alpha}(v) = \gamma_G(v + \alpha) - \alpha \).

The Minkowski and \( l_2 \)-Voronoi normal forms coincide if the basis is orthogonal with respect to the scalar product. If a basis is “almost orthogonal”; the Minkowski and Voronoi normal forms may coincide too, on all elements or on a controlled subset; the Minkowsky normal form is easy to compute if one knows the basis. The Gröbner normal form is (relatively) easy to compute if one knows the ordering and the Gröbner basis of the lattice with respect to the ordering. The problem is that the Gröbner basis is often very large, and its size may depend substantially on the ordering.

An element of \( \mathbb{Z}^n \) is said to be **standard** if it coincides with its normal form. The cardinality of each standard set is of course the same, being the cardinality of the quotient, and it is equal to the absolute value of the determinant of a matrix whose columns are a basis of the lattice.

### 2.3 Lattice cryptosystems, GGH

Many cryptosystems based on lattices have been defined [17], but most have been broken, and/or are impractical, they are hence a good company for Polly Cracker systems; their attractive feature is efficiency, something that Polly Cracker can hardly claim. We will consider two such systems, the Goldreich-Goldwasser-Halevi (GGH) cryptosystem [8] in this section, and the NTRU cryptosystem [10] in the next one.

Lattice cryptosystems try to exploit the NP-hardness of the SVP (shortest vector problem) and CVP (closest vector problem) for lattices. SVP and CVP for the \( l_1 \) norm can be easily reduced to the computation of a Gröbner basis (SVP, approximated up to a factor of 2) or to the computation of the Gröbner normal form (CVP), with respect of a term-ordering compatible with the metric;\(^1\) hence it is cryptographically justified to try and use the Gröbner normal form in a lattice, in a way similar to GGH.

\(^1\)Any binomial corresponds to a vector that has a \( l_1 \) norm that is the sum of the degree of the head and of the tail. Since the ordering is degree-compatible, the degree of the binomial is equal to the degree of the head, and is at least one half of the \( l_1 \) norm, hence the smallest polynomial has at most twice the \( l_1 \) norm of the \( l_1 \)-smallest vector in the lattice.

The normal form of an element \( \alpha \) is the smallest positive vector equivalent to \( \alpha \). To remove the positivity assumption, add to every variable \( x_i \), a variable \( y_i \), and equations \( x_i y_i = 1 \); the normal form with respect to this ideal identifies the shortest vector equivalent to \( \alpha \), hence \( \alpha \sim \gamma(\alpha) \) is the closest vector.

This implies, of course, that computing the Gröbner normal form for a degree-compatible ordering in a full-rank integer lattice is NP-hard.

The GGH cryptosystem can be summarized as follows:
- Choose \( V \) as \( \{ s \cdot v \}_{s \in \mathbb{N}} \) for a suitable \( s \in \mathbb{N} \).
- Choose \( L \) through a basis \( B \) such that the elements of \( V \) are in Minkowski normal form; \( B \) is the private key.
- Choose \( B' \) another basis of \( L \) as public key. Heuristics are given to ensure that the elements of \( V \) cannot be reconstructed from \( \mu_B \).
- A message is an element \( m \in L \), and a corresponding cryptogram is an element \( c \in m + V \), obtained obscuring \( m \) through a random element \( v \in V \).

A cryptogram \( c = m + v \) is decyphered computing its normal form \( \mu_B(c) = v \), and computing \( l = c - \mu_B(c) \).

Any basis \( B'' \) can be used to decypher, provided that most elements of \( V \) are in Minkowski normal form, e.g. if \( B'' \) is “almost” orthogonal. The conjectured security of the cryptosystem relies on the difficulty of finding a basis that is almost orthogonal from a suitably generated \( B' \).

The positions of \( v \) and \( m \) can be exchanged: take \( v \) as message, and add a random obscuring element in \( L \). Clearly, retrieving \( v \) one can find \( l \) and conversely, so the two variants are equivalent. See [15] for more details.

This remark however (see also [15]) means that, when the message is given by \( v \in V \), if \( m + v \) is sent, \( \mu_B(v) = \mu_B(m + v) \) is easily computed. This shows that the randomization effect of \( m \) is completely illusory: the cryptosystem is as (in)secure as the one in which the encyphering is done as \( c = \mu_B(m) \), without any randomization.

Our description is not how the system was presented in [8], where it is stated that decyphering is done through the Voronoi normal form, but since \( B \) and \( V \) are chosen in such a way that they are the same, the two descriptions are equivalent.

GGH has been broken [16] using the fact that all the elements of \( V \) are congruent mod \( 2 \cdot \mathbb{Z}^n \); this can be fixed taking a different \( V \), but overall the choice of parameters is so fragile that it was impossible to fix it maintaining the practicability of the cryptosystem.

The variant presented in [15] specifies for \( B' \) the Hermite Normal Form (HNF), hence reducing considerably the length of the public key and the complexity of encyphering. Since the HNF can be computed in polynomial time, this variant is as secure as the original form is.

### 2.4 GB-GGH aka LPC

Using the Gröbner normal form instead of the Minkowski normal form for decyphering, and a different set \( V \), we define a cryptosystem that is both a variation of GGH and a *bona fide* Binomial Barkee cryptosystem, i.e. a Lattice Polly Cracker. This could be called either GB-GGH, BinBarkee or LPC (the last being our choice).

Assume that we have a lattice \( L \), a Gröbner basis \( G \) of the lattice with respect to a term ordering \( \tau \), and a set \( V \) of elements in Gröbner normal form. Let \( B \) be a basis of the lattice. We define the LPC cryptosystem as follows:
- The private key is \( G, \tau, L; \)
- The set of messages is \( V; \)
- The public key is \( B. \)
- A message \( m \) is encyphered as \( c = \mu_B(m); \)
- \( c \) is decyphered as \( \gamma_G(c) \) if \( \gamma_G(c) \in V \), otherwise it is decyphered as ERROR.
One of the main difficulties in defining a LPC instance is to decypher one has to compute a Gröbner basis; in computing it, a term ordering has to be chosen, and the complexity varies wildly depending on the ordering: we will see later that a Lex Gröbner basis of a full-rank lattice of dimension \( n \) has \( n \) elements and can be computed in polynomial time, while the Gröbner basis with respect of a degree-compatible ordering of a generic full-rank lattice has exponentially many elements (the doubly-exponential lower bound [14] can only happen for lattices that are not of full rank).

The definition of \( V \) rules out the possibility of using a Lex basis (for suitable choices of \( V \) the elements are not in Lex Gröbner normal form).

This means that the lattice \( L \) should not be generic, and the ordering should be tailored to \( V \) and \( L \); but the public presentation \( B \) of the lattice should efficiently conceal the information on the structure of \( L \) that could otherwise allow the computation of a Gröbner basis suitable for decyphering by an eavesdropper.

Remark that, while a Gröbner basis and \( \gamma_G \) require to fix a coordinate system to represent polynomials, \( \mu_B \) is independent of the coordinates; publishing \( V \) however requires a coordinate system, not necessarily the same that is used to define \( G \).

We will start discussing the case in which the coordinates are the same, will show that in this case the cryopystem is insecure, and finally propose a setting in which a change of coordinates makes the system more secure. The conclusion is that an attack to retrieve the change of coordinates might break the cryopystem, and we study how to make this attack hard to complete.

The set \( V \) should be chosen to avoid to disclose information on the staircase of \( G \) and on the private coordinates.

The natural choice is to define \( V \) as the set of monomials in which the degree of each variable is limited by a uniform bound \( a \), i.e. the set \( S_a = \{0, 1, \ldots, a\}^n \), or a subset, to avoid the possibility of some attacks. \( V \) of course has to be defined in the public coordinates, but has to be used in the private coordinates. This is the main difficulty in the definition of the cryopystem.

We need that \( G \) have the property that every element of \( V \) is in \( \gamma_G \) normal form. If there is no change of coordinates, this means that there should be no element of \( G \) whose leading term has all the variables with degree \( \leq a \).

If a change of coordinates is performed under some restrictions we can still safely bound \( V \). We define the maximal inner width \( MIW(G) \) and minimal outer width \( MOW(G) \);

\[
MIW(G) = \min_{g \in C}(\max_{\deg_g}, L(g)) \quad \text{and} \quad MOW(G) = \max_{g \in C}(\max_{\deg_g}, L(g))
\]

respectively. Large \( MIW \) allows to use \( G \) in encyphering, small \( MOW \) allows to use \( G \) for signatures.

LPC with a trapdoor lattice as described below is a Polly Cracker cryopystem that appears to be resistant to the known message attacks. The question is whether we can build a system that resists also to the attacks to the trapdoor information. This requires a change of coordinates, that might in turn require the use of a shifted Gröbner normal form.

We remark that any Gröbner basis \( G' \) with respect to any term ordering and any coordinate system can be used for decyphering, as long as \( V \) is composed of elements in \( \gamma_{G'} \) normal form, e.g. as long as \( MIW(G') \) is sufficiently large.

### 2.5 Signing through LPC

The decyphering procedure for a lattice ideal can be used to design a signature scheme, following the usual scheme of proving the signer identity showing that he can decypher a random message generated by a hash key. See also [8, 18].

There is a small asymmetry between decyphering and signing, caused by the need of concealing the shape of the staircase of the lattice ideal.

Let us recapitulate the situation. The public key allows to compute the Minkowski normal form \( \mu(x) \) of any element \( x \), and the private key allows to find the Gröbner normal form \( \gamma(x) \) of any element \( x \).

For encyphering, one has to specify a subset \( M \subseteq \mathbb{Z}^n \) such that \( x \in M \Rightarrow \gamma(\mu(x)) = x \). This means that \( M \) has to be a subset of the elements in Gröbner normal form.

For signatures, one has to commit to decypher (i.e. find an equivalent element in a prescribed region) any element \( x \); one has to choose a subset \( S \subseteq \mathbb{Z}^n \) where the signature might lie, hence \( S \) has to be a superset of the elements in Gröbner normal form.

One cannot take \( M = S = Can(I) \), the set of standard monomials of the private Gröbner basis, since this would reveal informations essential to the security.

One can instead take for \( M \) (resp. \( S \)) a subset (a superset) of power products in which the degree of every variable is upper (resp. lower) bounded by the \( MIW \) (resp. \( MOW \)).

We concentrate the discussion on encyphering, and only a few details on signature will be given.

### 2.6 Trapdoor lattice ideals

In a lattice Polly Cracker, (as in any other cryopystem) there are two factors to consider: how to protect the private key, and how to protect the message.

The private key, that is a Gröbner basis used to decypher a message, should be of moderate size, and easily computable. The knowledge of the lattice is not sufficient to produce the Gröbner basis: one has to choose the ordering, and start from a suitable set of generators (a basis of the lattice) and of coordinates.

Because of the lattice description, standard lattice tools can be used, notably the Lenstra-Lenstra-Lovász lattice reduction algorithm [12] and Schnorr’s modifications BKZ [19, 20]. These tools, when the dimension is small, can bring a basis of a lattice to a standard form, that is often convenient for some attacks, possibly after choosing a suitable scalar product.

This implies that, to hope to be secure, the dimension of the lattice, hence the number of variables in the associated polynomial ring, has to be large, probably more than 100. Gröbner bases of binomial ideals in hundreds of variables are extremely large (at least exponential in the number of variables) unless a very special structure is present, in the ideal and/or in the ordering. This structure may make the Gröbner basis manageable, either in general or just for some term ordering. For example, if an ideal is a sum of ideals generated by polynomials in disjoint sets of variables, then the Gröbner basis will be the union of the Gröbner bases, hence could be manageable with any term ordering.

We will construct ideals such that the Gröbner basis is relatively small only in a very restricted set of term orderings and coordinates, and use this set as part of a trapdoor. A set of coordinates is the other part of the trapdoor.

While the protection given by a basis of the lattice is only
2.7 Twisted product lattices

Given lattices \( L_1 \subseteq Z^{n_1}, L_2 \subseteq Z^{n_2} \) of full rank, a \( L \subseteq Z^{n_1} \times Z^{n_2} \) is called a twisted product of \( L_1 \) and \( L_2 \) if the projection of \( L \) into \( Z^{n_1} \) is \( L_1 \) and the intersection of \( L \) with \( Z^{n_2} \) is \( L_2 \). The ordering in \( Z^{n_1} \times Z^{n_2} \) is a product ordering if any element \((v_1, v_2)\) is positive whenever \( v_1 \) is positive. We will show below how a Gröbner basis of a twisted product lattice can be computed.

A twisted product lattice has a basis being the columns of a block matrix \( \begin{pmatrix} A_1 & 0 \\ C & A_2 \end{pmatrix} \); \( A_1, A_2 \) are bases of \( L_1 \) and \( L_2 \) respectively. This basis will be private, and the private coordinate system will be chosen compatibly with the product structure; the public coordinate system will hide this structure, as we will see later.

Every full rank lattice of dimension larger than 1 allows a decomposition as twisted product, (e.g. reducing a basis into Hermite normal form), but a random decomposition is unlikely to be useful for the purpose of decyphering messages.

We show that if \( A_1', A_2' \) are matrices that describe Gröbner bases of the lattices \( L_1, L_2 \), then one can build a matrix \( \begin{pmatrix} A_1' & 0 \\ C' & A_2' \end{pmatrix} \) describing a Gröbner basis of \( L \).

In order to simplify the description, we assume that \( A_1 \) and \( A_2 \) contain a column whose entries are all positive. This is always possible, because \( L_1, L_2 \) are of full rank, the quotient is finite, hence the lattice contains a vector whose coordinates are all positive. In this case, the Gröbner basis of the lattice is the Gröbner basis of the ideal generated by the columns of the matrix.

Consider a matrix \( M \) such that \( A_1 M = A_1' \). Let \( C'' = CM \). Then \( \begin{pmatrix} A_1 \end{pmatrix} \) defines the same lattice as \( \begin{pmatrix} A_1' \end{pmatrix} \).

Now we can find a matrix \( M' \) such that \( C' = A_2M' + C'' \) has all entries that are non-positive. The matrix \( \begin{pmatrix} A_1' & 0 \\ C' & A_2' \end{pmatrix} \) once again generates the same lattice of \( \begin{pmatrix} A_1 \end{pmatrix} \) \( \begin{pmatrix} C \end{pmatrix} \). We want to show that this matrix represents a (non reduced) Gröbner basis of the lattice.

Since \( C' \) is non-positive, the leading terms of \( \begin{pmatrix} A_1 \end{pmatrix} \) \( \begin{pmatrix} C' \end{pmatrix} \) \( \begin{pmatrix} A_2 \end{pmatrix} \) are the same as the leading terms of \( A_1' \), because the variables of the second block are infinitely small with respect to the variables of the first block; hence the multiplicity of the ideal generated by the leading terms of the binomials of the columns of \( A_1' \) is equal to the determinant of \( A_1 \) (that is non zero). The same for \( A_2 \). The argument in [21] shows that the binomials are a Gröbner basis: adding a non redundant element will decrease the multiplicity, that is bound to be equal to the determinant, hence cannot be further reduced.

Remark that the Gröbner basis that we constructed may be non reduced. Reducing it is however simple, being equivalent to reducing the tail of every basis element.

Remark also that \( MIW(L) = \min(MIW(L_1), MIW(L_2)) \) and \( MOW(L) = \max(MOW(L_1), MOW(L_2)) \). All the arguments of course extend to twisted products of several lattices.

2.8 Attacks to twisted product lattices

We show how to disclose a twisted lattice structure if the coordinates are known up to the order (shuffled but not changed). The attack often allows to find a Gröbner basis sufficiently similar to the original.

Remark that to break the private key it is not necessary to recover exactly the twisted product structure, i.e. the original decomposition. Any basis of the lattice would be OK provided that we a) can compute the Gröbner basis of the lattice with respect to an ordering; b) the basis computed is able to decypher a message, i.e. all the elements of \( V \), or at least a non irrelevant part of them, are in Gröbner normal form.

Indeed, any lattice in any coordinate system allows a twisted product decomposition: it is sufficient to compute the column Hermite normal form; an invertible matrix in Hermite normal form is a block decomposition in blocks of size 1. However, when in Hermite normal form, the determinant is the product of the diagonal elements. Quite often, many diagonal elements are 1, and this will certainly happen if the original matrix has a determinant that is a product of few primes, e.g. if the original square matrices have prime determinant. But this also happens if the matrices are random.

Using the twisted lattice decomposition given by a Hermite normal form has usually \( MIW = 1 \), and is hence unsuitable.

We have thus discovered an important fact: since a matrix in Hermite normal form is in twisted lattice form, with blocks of length 1, the Lex Gröbner basis can be straightforwardly computed, bringing with column elementary operation every element off the diagonal to be negative.

In particular, assume that the lower right corner of the matrix is \( m \), and let \( z \) be the corresponding variable; this means that \( z^{m} - 1 \) is an element of the reduced Gröbner basis of the lattice, and this means that \( z \) is an element of order \( m \) in the multiplicative group of \( k[X]/J \), where \( J \) is the lattice ideal. This remark allows to compute the multiplicative order of any variable, just putting the variable as last one, and computing an Hermite normal form. This method can be extended to any vector, through a change of coordinates.

More in general, from a basis of a lattice, one can compute the determinant, and the order of any element will be a divisor of the determinant.

i.e. a partition of the variables is defined, and comparison between monomials in different elements of the partition is lexicographic.
Assume that we know that a lattice is a twisted product \( L_1 \times L_2 \). Then the elements of the second block will have order that divides the determinant of the block, while the generic elements have order that divides the determinant of the lattice. In genericity conditions, this will allow to detect the block decomposition.

When the blocks of a lattice are identified, the attacker is in the position of finding a Gröbner basis of the lattice, with respect to a block ordering corresponding to the twisted product decomposition, exactly in the same position of the designer of the private key.

Protecting the block decomposition making it unreadable from the orders of the variables seems difficult, hence we need to protect the coordinates in which we have the original decomposition. This means that we have to publish the lattice in a basis that is different from the basis in which the private computations are performed.

To fix the notation, let \( A \) be the block matrix, whose columns are a basis of the twisted lattice. Changing the basis, i.e. disclosing \( AR \), \( R \) being a unimodular matrix, is not sufficient, we need to multiply it to the left too, by a unimodular matrix \( S \), obtaining \( SAR \). We will moreover need a shift vector \( s \). This vector should be such that if \( m \in V \) then \( Am + s \) is in Gröbner normal form.

Consider now the enciphering of a message \( m \in V \); it is enciphered as \( c = m + By \), \( y \) being a suitable vector with integer coordinates, so that \( By \) is a lattice element. To recover \( m \) one computes the normal form of \( S^{-1}c + s = S^{-1}m + Afy \); \( Afy \) is in the private lattice, and reduces to zero, hence if \( S^{-1}m + C \) is in normal form, then \( m \) can be retrieved through \( S(y(S^{-1}c) - s) \).

The highlighted condition shows that we cannot choose a generic unimodular \( S \), we need to choose it carefully together with \( V \) and \( s \). In practice, it turns out that the construction of \( S \) is more difficult than the construction of the twisted lattice, since the security of the cryptosystem relies on \( S \), hence one can choose \( S \) and \( V \) first, choose \( s \) such that \( S^{-1}V + s \) is in the positive quadrant, then build the private lattice in such a way that \( S^{-1}V + s \) is in Gröbner normal form.

The tuning of the construction might be delicate, since a matrix \( S \) that is too sparse allows attacks to the private key through the Hermite normal form, while a generic matrix \( S \) might allow attacks to the message through LLL and Minkowski normal form.

One still has to choose the matrix \( R \). Following the advice of [15], we might represent the lattice in (column) Hermite normal form. However, we rather propose to represent it in a slightly different form, defined as the Hermite normal form, but in which the elements off the diagonal are always negative instead of positive, and their absolute value satisfies the same bound defining the Hermite normal form. In this form, the basis of the lattice is a reduced Gröbner basis for the Lex ordering, and with respect to this basis the Minkowski normal form and the Gröbner normal form coincide, as it is easy to check. The difference is purely aesthetic: we have a cryptosystem in which the secret key and the public key are Gröbner bases of the same lattice, and enciphering and decyphering are made with the same algorithm.

2.9 Twisted product lattices for Polly Cracker

The choice of lattices for the building blocks of Polly Cracker has to be tuned considering efficiency and security. Up to now, we have strived for feasibility, making some reasonable assumptions and some experiments. These conclusions might be modified after closer security considerations.

The Gröbner basis of a lattice of full dimension in \( n \) variables with size of the random entries about 20 with DegRevLex ordering has size bounded (experimentally) by \( 3^n \), and usually one is not far from the limit. The computation is usually unfeasible for \( n > 12 \); \( n = 8 \) instead is fairly rapid, and the Gröbner basis has usually 4000–5000 elements.

With these lattices a MIW of 20, and a MOW of 100 are easy to obtain.

Exhaustive search attacks can be performed to the highest and lowest block, so it is reasonable to protect them with a larger size, while for the intermediate blocks a size of 4 (that usually gives a Gröbner basis of size 30; sizes larger than 50 are unusual) seems instead reasonable. The size of the exponents is of course bounded by the MOW.

A total dimension of the lattice of 128 might be good, although as low as 32 and as high as 512 do not differ much in size of the resulting Gröbner basis, since the bulk of it is given by the 10000 elements of the cornerstones; adding 120 or 4000 elements for the middle blocks is not that much different. The main difference is given by the increase of the intermediate exponents of the Gröbner reduction, but this too can be tamed.

Of course, reducing a monomial by a basis of 10000 elements is not trivial. We have developed ad-hoc algorithms, making special use of the elements with pure power head, that are rather efficient; we can perform a reduction (i.e. a decyphering) in less than 1 second, with an unoptimized prototype, and we hope to improve it at least 100 times with a dedicated implementation. We don’t have (yet) a reasonable complexity estimate on decyphering.

We have concentrated on enciphering, it seems possible that for signatures the size of the private key can be considerably reduced, since a partial Gröbner basis is sufficient for signing, hence a private key of \( n^2 \) bytes seems possible. The complexity of signing does not seem however to be radically improved with the partial basis. More technical details on the choice and its reasons will be contained in a forthcoming technical paper.

3. IMPROVING NTRU

3.1 Introduction

In this section we examine another cryptosystem, NTRU [10] that can be attacked with lattice methods. NTRU is industrially used, but is considered at least with suspicion for a series of weaknesses. We will show that with the use of multivariate polynomial algebra, and of two Gröbner bases that are part of the private key, one can design an improvement that considerably strengthens the cryptosystem without seriously degrading the performance. In particular, the known lattice attacks to the private key are made impossible since now the key construction has a trapdoor protected by a private (instead of a public) lattice, and the attacks to the message are made more difficult since the message now can have larger norm.

3.2 Description of NTRU

NTRU [10] is not a lattice cryptosystem, it is a polynomial algebra cryptosystem allowing a lattice interpretation.
The reason is that if $\phi$ is a monic polynomial in $\mathbb{Z}[X]$ of degree $n$, then $A = \mathbb{Z}[X]/(\phi)$ is isomorphic (as $\mathbb{Z}$-module) to $\mathbb{Z}^n$, and ideals of $A$ can be seen naturally as sublattices of $\mathbb{Z}^n$.

In the case of NTRU, $\phi = X^n - 1$. This means that multiplication by $X$ is a cyclic permutation of the canonical basis of $\mathbb{Z}^n$, and that an ideal is a lattice invariant by this action of the cyclic group.

The setup of NTRU is given by $n$, two elements $q, p$ of $A$ that are coprime (i.e. that they generate $(1)$ in $A$); to simplify the description, $q$ is a prime number in the range 100-500 and $p$ is 2 or 3, but different choices are possible.

A polynomial in $\mathbb{Z}[X]$ is called small if its coefficients are at most $p/2$ in absolute value, and its support is “small” (the bound depends on the setting and the context, it may be different for polynomials used differently); it is called moderate if its coefficients are smaller than $q/2$ in absolute value.

The private key is composed of two small polynomials $f$ and $g$, both invertible mod $q$ and with $f$ also invertible mod $p$. The inverses are $f^{-1}$, $g^{-1}$ respectively (the inverse of $g$ is never used). The public key is $h = g f^{-1}$. A message is a small polynomial $m$, and is encyphered into a cryptogram $c = ph + m \mod q$, where $r$ is a random small polynomial. The parameters are chosen to ensure with high probability that $g pr + fm$ is moderate.

Decyphering is done as follows: compute $fc \equiv g pr + fm$ and represent it mod $q$ as a moderate polynomial $\phi \in A$. Assuming that $g pr + fm$ is moderate (and this is at least probabilistically granted by the size of the support) then $\phi \equiv fm \mod p$, hence $f^{-1} \phi \equiv m \mod p$, and because $m$ is small it is recovered from its value mod $p$.

If $p = 2$ the only change needed is that the message $m$ is a polynomial with coefficients in $(0, 1)$ but, before encyphering, the signs are randomly assigned.\(^3\)

### 3.3 NTRU Lattice attacks

NTRU can be attacked interpreting its operations in a sublattice of $A \times A$, see [10, 4]. Consider the sublattice $L$ of $A \times A$ generated by $(q, 0)$ and $(h, 1)$. This lattice has dimension $2n$ and determinant $q^n$, hence the shortest vector has norm comparable to $\sqrt{n}$, and because of the symmetry there is generically a space of dimension $n$ of vectors of minimal length.

Since $fh \equiv g \mod q$, the lattice contains $(g, f)$, that is (because of the smallness of $f$ and $g$, required to make decyphering highly probable) an element of minimal norm in the lattice $L$. Lattice reduction algorithms are likely to find $(g, f)$, and indeed some early variants with smaller length have been broken with this method.

This attack, if successful, recovers either the private key, or an equivalent key; a similar attack can be attempted to recover a message, since $(ph, r)$ is the lattice closest vector (under a suitable norm) to $(c, 0)$. In turn, it is likely that the closest vector problem can be solved through the shortest vector problem in the sublattice of $A \times A \times \mathbb{Z}$ generated by $L$ and $(c, 0, 1)$. The shortest vector itself is assumed to be $(m, r, 1)$.

The attack does not work in practice for the suggested parameters, because, fine-tuning them, the shortest vector

\(^3\)Thy need to be deterministically assigned if one wants to reconstruct $r$; in our extension this is needed to avoid some chosen cyphertext attacks.

of $L$ is not much shorter than the vectors in a large subspace, and the standard algorithms in that case either do not work or have high complexity.

One can remark that the security of the private key relies on having $f$ and $g$ of sufficiently large norm, the security of the message relies on having $r$ and $m$ sufficiently large, but if they are too large then $pr + fm$ is no longer moderate, hence the message cannot be decyphered. If one has to preserve decyphering, strengthening the key weakens the message, and conversely. The path to preserve both is narrow, and might require to increase $n$, hence message length and encyphering/encyphering costs.

### 3.4 GB-NTRU

Clearly, it is possible to use multivariate polynomials instead of univariate in NTRU. This gives no substantial advantage, but is essential for our extension. Because of the dimension of $A$ it is unreasonable to use more than 4 variables, and reasonable examples with more than 2 variables are exceptional, so we limit the description to 2 variables.

The ring $A$ will be $\mathbb{Z}[X, Y]/(X^n + 1, Y^{n_2} + 1)$. In our extension we will need $n_1 = n_2$, so to simplify notations we take directly $n = n_1 = n_2$. Everything else is unchanged, except that the lattice is invariant under a group that is not cyclic.

One might also consider polynomials different from $X^n - 1$, but, with the exception of $X^n + 1$, multiplication by $X$ is not an isometry, and this would just make decyphering much more difficult.

The basic idea of our extension, that we call GB-NTRU, is to use a different lattice for encyphering and decyphering; more precisely, we define an ideal $I \supseteq (X^n - 1, Y^{n_2} - 1)$ and keep it secret. It will be used for key creation and decyphering, while the public key is undistinguishable from a (bivariate) NTRU public key, and encyphering is performed in the same way.

Disclosing the private ideal will allow the classic attacks to NTRU, hence it is vital that exhaustive search for ideals containing $(X^n - 1, Y^{n_2} - 1)$ in $\mathbb{Z}[X, Y]/q$ is infeasible; and maximizing the number of such ideals just means that $x^n$ splits in linear factors in $\mathbb{Z}/q$, i.e. $n$ should divide $q - 1$.

This condition, combined with the condition that $q$ be smaller than the length of messages (that will be used later) shows that an univariate GB-NTRU is inferior, as well as a bivariate GB-NTRU with $n_1 \neq n_2$.

#### 3.4.1 Choosing the private ideal

Under the condition that $q$ is prime and $n \mid (q - 1)$, any ideal of $\mathbb{Z}[q][X, Y]$ containing $(X^n - 1, Y^{n_2} - 1)$ corresponds to a subset $S$ of the set of points $(x, y)$ such that $x^n = y^n = 1$. A point $(x, y)$ corresponds to the ideal $I(x, y) = (X - x, Y - y)$ of the polynomials vanishing in $(x, y)$, and $S$ corresponds to $\bigcap I(x, y) \in S I(x, y)$; moreover $(X^n - 1, Y^{n_2} - 1)$ itself corresponds to the set $S_n$ of all solutions of $(X^n - 1, Y^{n_2} - 1)$. We will rather use a complementary notation, indicating $J_T = I_{S_n \backslash T}$.

The ideal $J_T(q, X^n - 1, Y^{n_2} - 1)$ is a vector space of dimension equal to the cardinality of $T$, and a basis is composed of $(X^n - 1)(Y^{n_2} - 1)/(X - x)(Y - y)$ where $(x, y) \in T$.

#### 3.4.2 Preparing the public key

The preparation of the public key is simple, but good working keys are not easy to find. If a key works well or
not has to be checked after the preparation, and depends on the sublattice of the ideal \( J_r \).

We choose random small polynomials \( f \) and \( g \) in \( A \) that are invertible mod \( J_r \). This means that \( f \) and \( g \) do not vanish on any point of \( S_n \setminus T \). The size of the support of \( f \) and \( g \) is for now left undetermined, and will be discussed later.

We check that \( f \) is invertible mod \( p \) too (the case in which this assumption is relaxed will be discussed later).

Although we have discussed \( T \) before \( f \) and \( g \), it is useful to choose \( f \) and \( g \) first, and \( T \) after them. The point is that \( f \) is invertible mod \( J_r \) if and only if any zero at a point of \( S_n \) is contained in \( T \), so choosing \( T \) after \( f \) and \( g \) makes more likely that \( f \) and \( g \) have roots, and this might be useful.

We now build \( f', g' \in A \). \( f' \equiv f \) and \( g' \equiv g \mod J_r \); this means that we add to \( f, g \) a random element of \( J_r \), and we impose that \( f', g' \) are invertible mod \( S_n \). Now \( f', g' \) are no longer small; we compute a moderate polynomial \( h \equiv f^{-1}g' \mod q \), and \( h \) will be the public key.

We have to check that the combination of the public key with the private key, consisting of \((T, f, g)\), allows deciphering.

Retrieving \( f', g' \) or \( f, g \) from \( h \) through a short element in the NTRU lattice is no longer possible, since \((g', f')\) is not short, and \((g, f)\) is not in the NTRU lattice: \((g, f)\) is in a larger lattice that is private, i.e. \( J_r \), and if \( T \) is sufficiently large, then exhaustive search is impossible.

One can always find a \( T' \) such that \( h \) can be factored as a product of small polynomials mod \( J_r \), but usually with \( T' \) too large to give a key allowing deciphering.

### 3.4.3 Encyphering

Encyphering is done like in NTRU: a message is a small polynomial \( m \), \( r \) is a random small polynomial and \( m \) is encyphered as \( c = pr + m \).

Now although we have the same encyphering of NTRU, it can be made more robust. We have remarked that the larger is the support of \( r \) and \( m \), the more difficult it is to break the message through the NTRU lattice. But in NTRU one cannot increase the support without either weakening the key or making deciphering failures more likely, or even deciphering almost always impossible. In GB-NTRU instead the strength of the public key lies in the private lattice, and the support of \( f \) and \( g \) can be kept very small, just what is needed to protect the private lattice from exhaustive search attacks. This allows to increase the support of \( r \) and \( m \) in such a way that \((gpr, fm, 1)\) has norm larger than \( \sqrt{q} \). Hence it is not a shortest vector in \((q, 0, 0), (h, 1, 0), (c, 0, 1)\).

This means that the attacks to the message will be harder than in the original NTRU, hence the same level of security might be obtained with shorter lengths.

### 3.4.4 Decyphering

The receiver now has \( c = ph + m \) and wants to find \( m \). He computes \( fc \equiv gpr + fm \mod J_r \), hence \( fc = gpr + fm + \alpha \), \( \alpha \in J_r \). \( \alpha \) is unknown to the receiver, since it depends on \( r \), and \( h' = gpr + fm + \alpha \) is not moderate (while \( gpr + fm \) is—conjecturally—moderate). We try to guess \( \alpha \), conjecturing that \( \alpha \) is the closest vector to \( h' \) of \( J_r \) (seen as lattice). If we have a “good” basis of \( J_r \), and this can be checked at key creation, the closest vector can usually be determined through the Minkowski normal form (Babai Round-off algorithm). If \( J_r \) is not sufficiently good, or anyway the deciphering fails (this means that \( h'' = h' - \alpha \), the candidate \( gpr + fm \), does not satisfy the bounds) one can use the Babai closest plane algorithm CPA, \([1]\), but this is usually much more expensive, and should be avoided if possible; hence only keys for which the need of using the CPA has experimentally low probability should be used.

From now on, everything proceeds like in NTRU: let \( h'' \) be the candidate computed above, reduce \( h'' \) mod \( p \), multiply by \( f^{-1} \); the result will be \( m \), unless some of the conjectures (the fact that \( \alpha \) is the closest vector and that \( gpr + fm \) is moderate) are false. But as in NTRU we can tune the parameters to ensure that the conjectures are true with high probability. At the end, it is advisable to reconstruct \( r \) and \( m \), and check that they satisfy the specifications. This might identify deciphering errors, and allow to avoid chosen ciphertext attacks.\(^4\)

### 3.5 When \( f_r \) is not invertible

In some case (for some choices of \( n \) and \( p \)) it is very difficult to ensure that a random polynomial is invertible mod \( p \). This happens in particular when the splitting field of \( X^n - 1 \) has low degree over \( \mathbb{F}_p \).

In that case, identify an ideal \( J' \subseteq A \) such that \( fJ' \subseteq pA \)—one can take \( pA : (f) \). Then \( f \) will be invertible mod \( J' \), the deciphering computations will be mod \( J' \) instead of being mod \( p \).

The deciphered polynomial is ambiguous, but restricting the support of \( m \) to a subset of monomials independent mod \( J' \), the ambiguity is solved. In practice, this means forbid the few largest monomials, and can be integrated in the protocol to avoid disclosing informations on \( f \).

### 3.6 Parameters for GB-NTRU

The need of having \( n \) prime, \( n|q \) and a preference for \( q < n^n \) to increase the message security considerably limits the set of acceptable \( n, q \) pairs.

We have experimented a few settings; we limited ourselves to two settings, that has proven to be reasonable, for extensive testing. We chose \( q = 131 \) or \( q = 157, n = 13, p = 2 \), \( |T| = 8 \), size of the support of \( f \) and \( g \) equal to 11 and of the support of \( r \) equal to 60. With these choices the density of successful keys is more than 1/1000. For each a large set of messages have been tested for robustness, and all have shown to resist the standard attacks.

While GB-NTRU does not improve NTRU with respect to provable security, it seems to allow shorter block lengths. Bivariate polynomial arithmetic in degree \( n \) has the same cost as univariate polynomial arithmetic in degree \( n^2 \), hence the size of the public key and the cost of encyphering only depend on the block length, that is reduced. There is some loss of performance because the arithmetic with respect a prime modulus \( q \) is marginally more costly than arithmetic modulo \( 2^m \), that is the suggested choice for \( q \) in NTRU, but this too is compensated by the shorter block length. Deciphering is however more costly, and we have not yet tested in deep the probability of deciphering failures, that are harder to analyze for GB-NTRU than for NTRU.

\(^4\)This is a sketch of a chosen cyphertext attack, and how reconstructing the random element used in encoding will counter it: let \( c \) be a cyphertext, and let \( \alpha = (X^n - 1)(Y^n - 1)/(X - a)(Y - b) \); then \( (a, b) \in T \iff c \) and \( c + \alpha \) are deciphered in the same way; this would allow to reconstruct the private ideal, testing all the points. If instead \( r \) is checked, \( c + \alpha \) is recognized as invalid.
5. BARKEE WAS WRONG. OR NOT? CONCLUSIONS.

We have shown that Gröbner bases can be used to build public key cryptosystems by building two cryptosystems.

The first, a lattice cryptosystem of the Barkee family (AKA Polly Cracker) apparently resists to all known attacks to Polly Cracker and to lattices, although it might be subject to different types of attacks, and notwithstanding the size of the private key can be considered practical. Unfortunately the decyphering is rather slow, and we do not have yet a bound on the complexity.

In the second, of a completely different type, the Gröbner basis is surprisingly used only in the private key, and improves a currently used commercial cryptosystem.

Can we conclude that Barkee was wrong?7

The second cryptosystem falsifies Barkee’s title, nor Barkee’s argument, and the security of the first example has not yet undergone a serious public scrutiny to decide if it will be another failure and Barkee was right, or if this or a better incarnation will show that Barkee was wrong. So now we expect the reactions of Barkee’s friends.5

6. REFERENCES


5 Boo Barkee died several years ago, old and full of days. This is not surprising, since he was a dog, and his paper was written 14 years ago.

We profit of this footnote to dispel some false rumors on Boo Barkee and his coauthors. These rumors said that Boo Barkee was dead, and we can confirm, but also said that he was a Greek general.

This false information probably originates from his address in Ithaca, NY; people probably misread the state. As far as Barkee’s co-authors are concerned, Deh Cac Can is known to sign right to left, Julia Ecks was closely connected with Boo Barkee, Moriarty is still struggling with his archenemy, and the anonomy of R.F.Ree (Miskatonic University, Arkham, MS) should forever be preserved.

[7] 4ti2 team. 4ti2—A software package for algebraic, geometric and combinatorial problems on linear spaces. Available at http://www.4ti2.de