Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases

Jean-Charles Faugère[1] and Antoine Joux[2]

[1] Projet SPACES LIP6/LORIA CNRS/UPMC/INRIA
[2] DCSSI/Crypto Lab, 51 Bd de Latour-Maubourg, 75700 PARIS 07 SP

Abstract. In this paper, we review and explain the existing algebraic cryptanalysis of multivariate cryptosystems from the hidden field equation (HFE) family. These cryptanalysis break cryptosystems in the HFE family by solving multivariate systems of equations. In this paper we present a new and efficient attack of this cryptosystem based on fast algorithms for computing Gröbner basis. In particular it was was possible to break the first HFE challenge (80 bits) in only two days of CPU time by using the new algorithm F5 implemented in C.
From a theoretical point of view we study the algebraic properties of the equations produced by instance of the HFE cryptosystems and show why they yield systems of equations easier to solve than random systems of quadratic equations of the same sizes. Moreover we are able to bound the maximal degree occuring in the Gröbner basis computation.
As a consequence, we gain a deeper understanding of the algebraic cryptanalysis against these cryptosystems. We use this understanding to devise a specific algorithm based on sparse linear algebra. In general, we conclude that the cryptanalysis of HFE can be performed in polynomial time. We also revisit the security estimates for existing schemes in the HFE family.

1 Introduction

The idea of using multivariate quadratic equations as a basis for building public key cryptosystems appeared with the Matsumoto-Imai cryptosystem in [18]. This system was broken by Patarin in [20]. Shortly after, Patarin proposed to repair it and thus devised the hidden field equation (HFE) cryptosystem in [22]. While the basic idea of HFE is simple, several variations are proposed in [22]. As of today, the security of HFE, especially with variations is not well understood. Some systems have been attacked, others seem to resist. However, it is hard to draw the boundary between secure and insecure schemes in the HFE family.

Among the known attacks, we can roughly distinguish two classes. The first class consists of specific attacks which focus on one particular variation and breaks it due to specific properties. Typical examples are the attack of Kipnis and Shamir against Oil and Vinegar [14] and the attack by Gilbert and Minier [13] against the first version of the NESSIE proposal Sflash. The second class of attack consists of general purpose algorithms that solve multivariate system of equations. In this class, we find the relinearization technique of [15], the XL algorithm [8] and also the Gröbner basis [1–3] approach.

The relinearization technique of Kipnis and Shamir [15] is well suited to the basic HFE schemes without variations. The first step is to remark that some part of the secret key is a solution of some overdefined quadratic system of equations, with ϵm^2 equations involving m variables. The relinearization technique can even solve random systems as long as they are "sufficiently" overdefined. In fact, the success of the relinearization cryptanalysis against HFE is evaluated by estimating its complexity and success rate against random systems of the same size.

Gröbner basis is a well established and general method for solving polynomial system of equations. Of course the efficiency of this attack depends strongly on the algorithm used for computing the Gröbner basis: the first HFE challenge broken in section 3.4 is completely out of reach of the Buchberger algorithm (the historical algorithm for computing Gröbner bases). With this second class of attacks, the main difficulty is, given a set of parameters from an HFE cryptosystem, to determine the attacks' complexity.

In this paper, we show that the weakness of the systems of equations coming from HFE instances can be *explained* by the algebraic properties of the secret key. From this study we are able to predict the maximal degree occuring in the Gröbner basis computation. This accounts well for the experimental results: we made a series of computer simulations on real size HFE problems (up to 160 bits) so that we can establish precisely the complexity of the Gröbner attack and we compare with the theoretical bounds.

We also devise a specific algorithm based on efficient sparse linear algebra algorithms over $\mathbb{F}_2$ such as Block Lanczos. Finally, in section 5.4 and 5.1 we give concrete security estimates for existing cryptosystems in the HFE family.

2 General description of multivariate cryptosystems

The basic idea of multivariate cryptosystems from the HFE family is to build the secret key on a polynomial S in one unknown x over some finite field (often $\mathbb{F}_{2^n}$). Clearly, such a polynomial can be easily evaluated, moreover, under reasonable hypothesis, it can also be "inverted" quite efficiently. By inverting, we mean finding any solution to the equation $S(x) = y$, when such a solution exists. The secret transformations (decryption and/or signature) are based on this efficient inversion. Of course, in order to build a cryptosystem, the polynomial S must be presented as a public transformation which hides the original structure and prevents inversion. This is done by viewing the finite field $\mathbb{F}_{2^n}$ as a vector space over $\mathbb{F}_2$ and by choosing two linear transformation of this vector space L_1 and L_2. Then the public transformation is $L_2 \circ S \circ L_1$.

By enforcing a simple constraint on the choice of S, this transformation can be be expressed as a set of quadratic polynomial in n unknowns over $\mathbb{F}_2$. The constraint on S is that all the monomials occuring in S should be either constant, of degree 2^t (for some t) or of degree $2^{t_1} + 2^{t_2}$. Indeed, if $x_1, \ldots, x_n$

are the coordinates of x in some basis of $\mathbb{F}_{2^n}$, we can express each coordinates of S over $\mathbb{F}_2$ as a polynomial in $x_1, \ldots, x_n$. In this expression into multivariate polynomials, the constant monomial from S is translated into constants, the monomials of degree 2^t are translated into linear terms and the monomials of degree $2^{t_1} + 2^{t_2}$ into quadratic terms. This is due to the simple fact that the Frobenius map on $\mathbb{F}_{2^n}$, i.e., squaring, is a linear transform over $\mathbb{F}_2$ of the vector space $\mathbb{F}_{2^n}$.

From this basic idea, we obtain a trapdoor one-way function. Indeed, the public transformation expressed in multivariate form as:

$$y = (p_1(x_1, \cdots, x_n), \cdots, p_n(x_1, \cdots, x_n)),$$

is easy to compute and seems hard to invert, while the secret expression in univariate form is easy to invert when S is correctly chosen. In cryptosystems based on HFE, S is usually a low degree polynomial, which can be inverted by using a general purpose root finding algorithm over $\mathbb{F}_{2^n}$. In some systems, such as Sflash, we encounter an alternative construction, quite similar to RSA, where S consists of a single monomial of degree $2^t + 1$, for some t. In that case, to find a $2^t + 1$-th root of a finite field element y, we proceed as for RSA decryption, i.e., we raise y to the power d, where d is the inverse of $2^t + 1$ modulo $2^n - 1$. However, in that case, the basic system reduces to the Matsumoto-Imai cryptosystem [18] which is not secure (see [20]) and thus, it should be strengthened by removing some public equations. This strengthening operation which can also be applied when S is a low degree polynomial is discussed in section 4.1. In the sequel, we focus on the case where the degree D of S is small. In this case, the performance of the cryptosystem is directly related to that of the root solving algorithm that inverts S in $\mathbb{F}_{2^n}$. To illustrate this performance, the following table reports the time needed to find one such solution using NTL[25] (PC PIII 1 Ghz):

(n, D)	(80,129)	(80,257)	(80,513)	(128,129)	(128,257)	(128,513)
NTL (CPU time)	0.6 sec	2.5 sec	6.4 sec	1.25 sec	3.1 sec	9.05 sec

From these data, we conclude that for the sake of speed, the degree D of the secret polynomial must remain quite small, say at most 512. For example, the recommended values for the first HFE challenge are $n = 80$ and $D = 96$. For this reason, throughout the rest of the paper, we focus on the study on HFE cryptosystems under the assumption that D remains fixed as n grows.

3 Gröbner basis cryptanalysis

We refer to [5, 9] for basic definitions. Let k be a field and $R = k[x_1, \ldots, x_n]$ the ring of multivariate polynomials. To a system of equations

$$f_1(x_1, \ldots, x_n) = \cdots = f_m(x_1, \ldots, x_n) = 0$$

we associate the ideal I generated by $f_1, \ldots, f_m$. A Gröbner basis is a generating system of I that behaves nicely with respect to some order on monomials. We recall that a monomial in $x_1, \ldots, x_n$ is a term in $x_1, \ldots, x_n$ together a coefficient. Let $<$ denotes, an admissible ordering on the monomials in $x_1, \ldots, x_n$. A frequentely encountered example is the lexicographical ordering which is defined as $x_1^{\alpha_1} \cdots x_n^{\alpha_n} < x_1^{\beta_1} \cdots x_n^{\beta_n}$ iff $\alpha_i = \beta_i$ for $i = 1, \ldots, k$ and $\alpha_k < \beta_k$ for some k. Then for each polynomial f in R we can define its leading term $\mathrm{LT}(f)$ (resp. its leading monomial $\mathrm{LM}(f)$) to be the biggest term (resp. monomial) with respect to $<$.

Definition 1. *G a finite set of elements of I is a Gröbner basis of $(f_1, \ldots, f_m)$ wrt $<$ if for all $f \in I$ there exists $g \in G$ such that $LT(g)$ divides $LT(f)$.*

Let K be a field containing k, we can define the set of solutions in K which is the algebraic variety:

$$V_K = \{(z_1, \ldots, z_n) \in K \quad | f_i(z_1, \ldots, z_n) = 0 \; i = 1, \ldots, m\}$$

which is in fact the set of roots of the system of equations. Gröbner bases can be used in various situation (for instance when the number of solution is infinite of for computing real solutions). In the case of HFE we want to compute solutions of algebraic systems in $\mathbb{F}_2$. The following proposition tell us how to use Gröbner bases in order to *solve* a system over $\mathbb{F}_2$:

Proposition 1. *The Gröbner basis of $[f_1, \ldots, f_m, x_1^2 - x_1, \ldots, x_n^2 - x_n]$, in $\mathbb{F}_2[x_1, \ldots, x_n]$, describe all the solutions of $V_{\mathbb{F}_2}$. Particular useful cases are:*

i) $V_{\mathbb{F}_2} = \emptyset$ (no solution) iff $G = [1]$.
2) $V_{\mathbb{F}_2}$ has exactly one solution iff $G = [x_1 - a_1, \ldots, x_n - a_n]$ where $a_i \in \mathbb{F}_2$. Then $(a_1, \ldots, a_n)$ is the solution in $\mathbb{F}_2$ of the algebaric system.

This proposition tell us that we have to add the “field equations” $x_i^2 = x_i$ to the list of equations that we want to solve. Consequently we have to compute a Gröbner basis of $m+n$ polynomials and n variables. In fact, the more equations you have the more able you are to compute a Gröbner basis.

3.1 Useful properties of Gröbner bases

Another order on monomials is the Degree Reverse lexicographical order or (**DRL** order). This order is less intuitive than the lexicographical order but it has been shown that the DRL ordering is the most efficient, in general, for computating Gröbner bases.

$x_1^{\alpha_1} \cdots x_n^{\alpha_n} >_{\mathrm{DRL}} x_1^{\beta_1} \cdots x_n^{\beta_n}$ iff $deg(x^\alpha) = \sum_{i=1}^{n} \alpha_i > deg(x^\beta)$ or $deg(x^\alpha) > deg(x^\beta)$ and, in $\alpha - \beta \in \mathbb{Z}^n$, the right-most nonzero entry is negative.

We have seen in proposition 1 that Gröbner bases are useful to solve a system but they can also be used to discover low degree relations:

Proposition 2. *If G is a Gröbner basis of an ideal I for $<_{\mathrm{DRL}}$ then G contains all the (independent) equations in I of lowest total degree.*

By computing Gröbner bases it is even possible to find *all* the algebraic relations among $f_1, \dots f_m$ (see [9] page 338 for a precise definition of the ideal of relations).

Proposition 3. *([9] page 340)*
Fix a monomial order in $k[x_1, \dots, x_n, y_1, \dots, y_m]$ where any monomial involving one of the $x_1, \dots, x_n$ is greater than all monomials in $k[y_1, \dots, y_m]$ (lexicographical ordering for instance) and let G be the Gröbner basis for this ordering. Then $G \cap k[y_1, \dots, y_m]$ describe all the relations among $f_1, \dots, f_m$.

By combining proposition 2 and 3 we can find the lowest degree relations among the f_i. This will enable us to give predict the computation of a Gröbner basis in the multivariate world from another (easy) Gröbner basis computation in the univariate world (see section 5.1).

3.2 Algorithms for computing Gröbner bases

Note that definition 1 does *not depend on a particular algorithm.* In the section, we quickly review existing algorithms and especially the recent improvements in the field. Historically the first algorithm for computing Gröbner basis was presented by Buchberger [1–3]. The Buchberger algorithm is a very practical algorithm and it is implemented in all Computer Algebra Systems (a non exhaustive list of efficient implementations is: Magma, Cocoa, Singular, Macaulay, Gb, ...); section 3.4 contains a comparison between them for the HFE problem. More recently, more efficient algorithms for computing Gröbner bases have been proposed. The first one F_4 [10] reduces the computation to a (sparse) linear algebra problem (from a theoretical point of view the link between solving algebraic systems and linear algebra is very old, e.g., see [17, 16]). More precisely the algorithm F_4 incrementally construct matrices in degree 2, 3, ... D:

$$A_D = \begin{array}{c} \\ m_1 \times f_{i_1} \\ m_2 \times f_{i_2} \\ m_3 \times f_{i_3} \\ \cdots \end{array} \begin{array}{c} \textit{momoms degree} \ \leq \ D \textit{ in } x_1, \dots, x_n \\ \begin{pmatrix} \quad\quad\quad\quad \cdots \quad\quad\quad\quad \\ \cdots \\ \cdots \\ \cdots \end{pmatrix} \end{array},$$

where $m_1, m_2, \dots$ are monomials such that the total degree of $m_j f_{i_j}$ is less than D. The next step in the algorithm is to compute a row echelon of A_D using linear algebra techniques. It must be emphasized that:

- The rows of A_D is a *small subset* of all the possible rows $\{m f_i \text{ s.t. } m \text{ any monomial } deg(m) \leq D - deg(f_i)\}$.
- The rows $m_j f_{i_j}$ in the matrix are not necessarily obtained from the initial set of equations $f_1, \dots, f_m$. Thus we can have $i_j > m$, in that case f_{i_j} was produced in a previous step of the algorithm (in degree $D' < D$).

The main drawback of this algorithm is that very often the matrix A_D does not have full rank. A new algorithm called F_5 that avoids this drawback have been proposed in [12]. This new algorithm replaces A_D by a full rank matrix, which is thus minimal (if the input system is regular see [12]). For the special case of $\mathbb{F}_2$ we use a dedicated version of this algorithm (called $F_5/2$) that also takes into account the action of the Frobenius on the solutions. Of course, the implementation of the linear algebra is also dedicated to $\mathbb{F}_2$. In the current implementation of this algorithm $F_5/2$ in the program FGb (written in C) we use a naive implementation of Gauss algorithm. The resulting algorithm/implementation is much more efficient than any other program for solving algebraic equations. We expect that by using sophisticated techniques described in section 5.2 the efficiency of such algorithm could be significantly improved.

From a complexity point of view the important parameters are: D the maximal degree occurring in the computation and N_D, the size of the matrix A_D. The overall complexity is N_D^ω where $2 \leq \omega \leq 3$ is the exponent of the linear algebra algorithm.

3.3 Complexity of Gröbner bases

The complexity of Gröbner bases algorithms has been studied in a huge number of papers. Over $\mathbb{F}_2$, when the "field equations" $x_i^2 - x_i$ are added, the set of solutions have a simple geometry. All the ideals are radicals (no multiple roots), zero dimensional (finite number of solutions). In fact it is easy to prove:

Proposition 4. *The maximal degree D of the polynomials occurring in the computation of a Gröbner basis including field equations $x_i^2 = x_i$ is less than n. The complexity of the whole computation is bounded by a polynomial in 2^n.*

Remark. Note that this result is only a rough upper bound. This must be compared with the complexity of the exhaustive search $\mathcal{O}(n2^n)$. In practice, however, efficient algorithms for computing Gröbner bases behave much better than in the worst case.

A crucial point in the cryptanalysis of HFE is the ability to distinguish a "random" (or generic) algebraic system from an algebraic system coming from HFE. We will establish in section 5.1 that this can be done by computing Gröbner bases and comparing the maximal degree occurring in these computations. As a consequence we have to describe theoretically the behavior of such a computation. This study is beyond the scope of this paper and is the subject of another paper [4] from which we extract the asymptotic behavior of the maximal degree occurring in the computation is: $d = \max \text{ total degree} \approx \frac{n}{11.114\ldots}$

From this result we know that computing Gröbner bases of random systems is simply exponential; consequently, in practice, it is impossible to solve a system of n equations of degree 2 in n variables when n is big (say $n \geq 80$).

3.4 First HFE Challenge is broken

The first HFE Challenge was proposed in [23] with a (symbolic) prize of 500$. This correspond to a HFE($d = 96, n = 80$) problem and can be downloaded from

Algorithm	Buchberger				F_4	F_5
System	Maple	Magma	Macaulay	Singular	FGb	FGb
CPU time < **10m**	12	17	18	19	22	**35**
CPU time < **2h**	14	19	20	21	28	**45**

Table 1. Comparison of various algorithms and implementations (PC PIII 1 Ghz)

[21]. For this problem, the exhaustive search attack require $\geq 2^{80}$ operations, hence is not feasible.

We have computed a Gröbner basis of this system with the algorithm $F_5/2$. As explained in section 3.2, the most time consuming operation is the linear algebra: for this example we have solved a 307126×1667009 matrix over $\mathbb{F}_2$. The total running time was 187892 sec ($\approx$ 2 days and 4 hours) on an HP workstation with an alpha EV68 processor at 1000 Mhz and 4Go bytes of RAM. Some care had to be taken for the memory management since the size of the process was 7.65 Giga bytes.

For this algebraic system [21], we found that there were is fact *four solutions*. Encoded as numbers by $X = \sum_{i=1}^{80} x_i 2^{i-1}$ the solutions are 644318005239051140554718, 934344890045941098615214, 1022677713629028761203046 and 1037046082651801149594670. It must be emphasized that this computation is far beyond reach of all the other implementations and algorithms for computing Gröbner basis as is made clear by the table 1. Because 80 equations of degree 2 was a previously untractable problem, this Gröbner basis computation represents a breakthrough in research on polynomial system solving.

4 Explanation of the algebraic cryptanalysis

In the cryptosystems described in section 2, the basic problem is to find solutions of equations of the form $y = L_2 \circ S \circ L_1(x)$, hidden in multivariate expressions. In section 3, we saw that general purpose Gröbner basis can be used to solve this problem. However, in general, computing such a Gröbner basis of a system of equations is quite difficult. We saw in section 3.4 that instances of HFE can be solved much faster than random systems of equations of the same size (in fact the complexity is only polynomial).

In order to explain this qualitative difference, we first need to show that the structure of the equations in an instance of HFE implies a relatively small upper bound on the degree m of the polynomials which occur during the Gröbner basis computation. In fact, we show in the sequel, that this bound depends on the degree of the secret polynomial S but not on the size of the field $\mathbb{F}_{2^n}$. On the other hand, with random systems, the degree of the intermediate polynomials strongly depends on n. Computing an exact bound on m is difficult, since the detailed behavior of general purpose Gröbner basis algorithms is quite complicated. Thus, we derive our bound by analyzing the basic linear algebra approach described in section 3. With this algorithm, we search for linear polynomials in the variables

x_i in the space of linear combination of multiples of the public polynomials p_i. When the degree of the intermediate polynomials is m, the degree of the multipliers is $m-2$. First, remark that a solution of $y = L_2 \circ S \circ L_1(x)$ is also a solution of $L_2^{-1}(y) = S \circ L_1(x)$ and that the degree of the multiples that need to be considered is thus independent of the linear transformations L_1 and L_2. As a consequence, it suffices to prove our statement for the equations directly derived from the secret equation $S - L_2^{-1}(y) = 0$. This can be done by working in the single variable description of S. Without loss of generality and to simplify the exposition, we assume the constant term is incorporated into S and look for a solution of $S = 0$. In that case, the multivariate equations associated to S can be expressed in term of S and its Frobenius transforms S^2, S^4, ... Moreover, low degree multiples of the multivariate equations can be obtained by multiplying S and its Frobenius transforms by monomials of the form x^d, where d is a sum of a small number of powers of two. More precisely, multiplying the multivariate equations by a monomial of degree t can be performed by multiplying S and its transforms by a polynomial whose monomials are of the form x^d, where the value of d is a sum of t (different) powers of two. Thus, it suffices to consider expressions of the form:

$$\sum_i \sum_j c_{i,j} x^j S^i, \tag{1}$$

where i is a power of two, j a sum of at most m powers of two, and $c_{i,j}$ are arbitrary elements in the field $\mathbb{F}_{2^n}$. Such an expression is linear as a multivariate polynomial, if and only if, it can be written as a polynomial in x whose monomials are either constant or x^{2^t} for some value of t.

With these remarks in mind, the original problem can be reformulated as follows: Find a linear combination of polynomials $x^j S^i$ (i power of two and j sum of at most $m-2$ powers of two), where all monomials x^d, where d is the sum of two or more powers of two, cancel out. Whenever such a combination exists it can, of course, be found through linear algebra. When, as in multivariate cryptosystems, the degree of S is bounded by D, we limit the degree[3] of the polynomials $x^j S^i$ we are considering by some bound H. Then, we count these polynomials, and denotes their number by N_P. We also count the number N_M of different monomials x^d, where d is a sum of at least two or more powers of two, appearing in all these polynomials. When N_P is larger than N_M, we have more polynomials to combine than terms to cancel out. Thus, unless some degeneracy occurs, the linear combination we want must exists. Moreover, the condition of degeneracy can be expressed algebraically by writing down some minor determinants that must vanish when a degeneracy is encountered. Thus, unless the (randomly chosen) coefficients of S satisfy these algebraic equations, we can guarantee the success of the attack. Clearly, the proportion of bad secret polynomials is exponentially vanishing in n. As a consequence, given D and

[3] When the degree of $x^j S^i$ is a power of two, the higher degree monomial correspond to a linear multivariate polynomial, as a consequence it can be ignored. In that case, we can slightly improve our bounds by replacing the degree of $x^j S^i$, by the degree of its second monomial. The datum provided in table 2 accounts for this fact.

m, it suffices to evaluate the numbers N_P and N_M for various values of H to determine whether Gröbner based cryptanalysis can attack an instance of HFE while considering intermediate polynomials of degree no higher than m. We do not explain any further how to solve this combinatorial problem. However, we now state some results and give some values N_P and N_M for many proposed parameters of HFE systems. It should be recalled that this results are upper bounds and than in practice, the attack may succeed even better.

The degree of the secret polynomial S is at least 3, otherwise, the public equations are entirely linear. For S of degree 3 and 4, it suffices to take $m = 3$, i.e., to multiply the public polynomial by the x_i variables and search for a linear combination. For degrees from 5 to 15, it suffices to take $m = 4$. From 16 to 127, $m = 5$ suffices. From 128 to 1280, $m = 6$ is enough. Finally, $m = 7$ works from degree 1281 up to degree 20480. These bounds really match the experimental complexity of the simple linear algebra approach except for the boundary cases of degree 16 and 128. On the other hand, more sophisticated Gröbner basis algorithms work even better.

The detailed results giving the relation between the degree of S and m, up to degree $D = 4096$, are shown in table 2. Note that some values for the degree of S are missing from this table, this is due to the simple fact that the degree of S is either a power of two or a sum of two such powers. Table 2 also gives the maximal number Max_{Eq} of independent equations that may be obtained by pushing the value of H even further.

4.1 Extension of the attack to HFE variations

In order to increase the security of HFE, several variations can be introduced. In particular, four simple variations are proposed in [22]. These variations are:

1. Remove some (multivariate) equations from the public key. Usually denoted by HFE-.
2. Add (useless) random equations to the public key. Usually denoted by HFE+.
3. Add extra variables with values in $\mathbb{F}_2$, usually denoted by HFEv. More precisely, the secret polynomial S is now a function of these extra parameters. However, the multivariate expression should remain quadratic. This is ensured by replacing the coefficient of "linear" monomials of the form x^{2^t} by an affine polynomial in the extra variables, similarly the constant term is replaced by a degree 2 polynomial in the extra variables.
4. Remove some variables by fixing their values. Usually denoted by HFEf.

All these variations can be combined in cryptosystems. The only restrictions are that variations 1 and 3 (HFE- and HFEv) are more suited to signature schemes and that variations 2 and 4 (HFE+ and HFEf) are more suited to encryption schemes. For example, when applying variation 1 to a signature scheme, we can remove a large number of equations from the public key, it will not prevent signature verification. On the other hand, with an encryption scheme, we can only delete a small number of public equations. Indeed, before performing decryption, it is necessary to guess the values of all deleted equations.

Degree D	m	H	N_P	N_M	Max_{Eq}	Degree D	m	H	N_P	N_M	Max_{Eq}
3	3	3	2	1	2	4	3	6	4	3	1
5	4	10	7	6	16	6	4	14	12	10	14
8	4	20	16	15	12	9	4	27	23	22	10
10	4	30	26	25	9	12	4	57	49	48	7
16	5	44	39	38	248	17	5	51	46	45	230
18	5	54	49	48	219	20	5	60	55	54	210
24	5	76	67	66	200	32	5	120	103	102	162
33	5	142	127	126	153	34	5	150	135	134	149
36	5	169	154	152	142	40	5	192	171	169	132
48	5	242	209	208	119	64	5	548	403	402	88
65	5	601	452	451	80	66	5	650	483	481	79
68	5	664	491	490	76	72	5	724	528	527	71
80	5	936	620	619	64	96	5	1856	1000	998	52
128	6	576	514	513	3763	129	6	584	527	526	3701
130	6	592	536	535	3671	132	6	616	558	557	3633
136	6	659	596	595	3550	144	6	689	620	619	3431
160	6	800	703	701	3285	192	6	977	829	828	3163
256	6	1665	1285	1284	2516	257	6	2060	1484	1482	2486
258	6	2070	1497	1495	2467	260	6	2090	1517	1515	2447
264	6	2147	1572	1571	2417	272	6	2320	1709	1708	2363
288	6	2470	1825	1824	2262	320	6	2828	2032	2031	2143
384	6	3633	2407	2406	1988	512	6	6528	3645	3643	1444
513	6	8337	4212	4211	1420	514	6	8408	4273	4272	1410
516	6	8784	4537	4536	1402	520	6	8896	4606	4605	1388
528	6	9234	4734	4733	1366	544	6	9520	4906	4904	1332
576	6	10378	5180	5179	1261	640	6	11792	5636	5633	1157
768	6	15616	6444	6443	1042	1024	6	74056	17376	17375	573
1025	6	82128	18487	18486	500	1026	6	82656	18689	18688	499
1028	6	83328	18858	18857	495	1032	6	84560	19069	19068	489
1040	6	85520	19139	19138	483	1056	6	89120	19438	19437	462
1088	6	152960	26249	26248	404	1152	6	180480	28270	28268	312
1280	6	246144	31102	31101	138						
1536	7	12805	8698	8695	41733	2048	7	19072	11702	11699	38779
2049	7	20487	12285	12284	38709	2050	7	20537	12441	12439	38688
2052	7	20563	12486	12484	38672	2056	7	20588	12510	12509	38648
2064	7	20652	12567	12566	38615	2080	7	20784	12669	12668	38571
2112	7	21072	12859	12857	38435	2176	7	21766	13211	13210	37882
2304	7	23072	13711	13709	36761	2560	7	25880	14484	14483	32707
3072	7	32512	16370	16368	31803	4096	7	71952	29494	29492	28394

Table 2. Upper bound on the efficiency of Gröbner cryptanalysis

In order to evaluate the impact of the variations on the security of HFE, we need to compute the new values of m (the maximal degree of multiplier monomials) when each variation is applied. The simplest cases are variations 2 and 4, which preserve m. Indeed, adding random polynomials as in variation 2 increases the number of columns in the matrix M but does not change the kernel we are looking for (since the number of lines in M is much larger than the number of columns). Similarly, fixing the value of some variables as in variation 4 collides some lines together, but leave the kernel unchanged.

The action of variation 1 (or HFE-) is more complicated. When a single equation is removed, the secret key view of the problem is changed. Instead of having an equation $S(x) = y$, we now have $S(x) = y_0$ or $S(x) = y_1$ depending on the value of the removed equation. These two possibility can be merged into the following equation (over $\mathbb{F}_{2^n}$):

$$(S + y_0)(S + y_1) = 0 \text{ or } S^2 + (y_0 + y_1)S + y_0 y_1 = 0.$$

Since this new equation is obtained by combining the Frobenius of S and S, it is again quadratic in the multivariate view. As a consequence, removing a single equation corresponds (at worse) to a doubling of the degree of S. Thus, the an upper bound on the value of m can once again be deduced from table 2. On the other hand, it should be noted that when some public equations are removed, the cryptanalysis can be somewhat improved. Indeed, if t equations are removed, then the system is underdetermined. Thus, we may guess the value of any set of t variables and with good probability still find a solution. Clearly, this slightly reduces the size of the linear algebra, since any line involving one of the t variables is either removed (when the variable value is 0) or merge with another line (corresponding to the monomial without this variable). With more sophisticated Gröbner basis algorithms the speed-up is quite noticeable.

The most complicated variation to deal with is variation 3 (or HFEv). In that case, in addition to the main variable x in $\mathbb{F}_{2^n}$, we have extra variables v_1, $\ldots$, v_s over $\mathbb{F}_2$. We assume here that s is small, which must be the case with encryption schemes and is usually true with concrete instances of signature scheme (typically, $s = 4$). In that case, any equation computed from S which is "linear" in the x_i may contains additional quadratic terms of the form $x_i v_j$ or $v_i v_j$. The problem for the cryptanalyst is that, after the outward linear transform, this simple structure is hidden. Luckily, there are in fact, many kernel equations coming from the low degree expression in x, and not a single one. With enough equations, we can carry the linear algebra step further on and remove the $x_i v_j$ and $v_i v_j$ terms. Indeed, we need to cancel the coefficients in v_1, $\ldots$, v_s before each "linear" term of the form x^{2^t}, we also need to cancel the coefficients of the $v_i v_j$. The total number of extra multivariate equations needed is $ns + s(s-1)/2$. Thus, for fixed s, it suffices to have $s+1$ linear equation in the univariate world to ensure that the attack again succeeds, without any change in the value of m.

5 Performance optimizations

5.1 Optimal degree with Gröbner basis algorithms

We have collected a lot of experimental data by running thousand of HFE systems for various $D \leq 1024$ and $n \leq 160$ (see [11]). In graph 5.1, the maximal degree m_{F_5} occurring in the Gröbner basis computation of an algebraic system coming from HFE (resp. from a random system as described in table 1 section 3.3) is plotted. This graph clearly shows that HFE algebraic systems do not behave as random systems since the maximal degree occurring in the computation does not depend on n the size of the field. The second point, is that the maximal degree m_{F_5} is always less than the upper theoretical bound m found in section 4; more precisely when $D < 513$, m_{F_5} is in fact equal to $m-1$ (except for $D = 16$ or 128) and when it is big enough $m_{F_5} = m$.

In section 4 we have derived an upper bound m by searching *linear* polynomials as a linear combination of "low degree" multiples of $S(x)$. By using again Gröbner bases and proposition 2 we can derive lower bound of complexity for

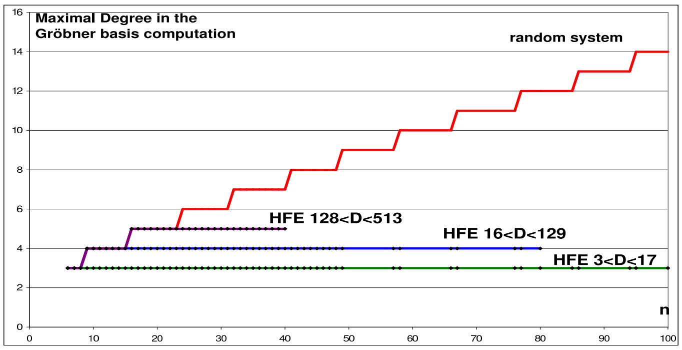

Table 3. Small dots correspond to a computer simulation.

any given instance of HFE. We search now *low degree* polynomials in the univariate world and not only linear relations. Let m^- be the minimal degree for which we have such a low degree relation. Of course, $m^- \leq m_{F_5} \leq m$. Another meaningful interpretation of m^- is that in this degree we can distinguish between an instance of HFE and a random system.

Since, we have to express that the degree of a monomial x^j where $j = 2^{i_1} + \cdots + 2^{i_k}$ is $w(j) = k$ (so that the degree is not j). First, we introduce new variables X_i corresponding to x^{2^i} together with the relations $X_i^2 = X_{i+1}$. Next, we define a "low degree relation" by saying that the monomials of highest degree vanish. To reach this goal, we introduce a new homogenization variable h so that all the variables X_i are bigger than h and we transform the equations into homogeneous equations; if we find in the ideal generated by these equations a polynomial f such that $f = hf'$ (in other words h is a factor of f) then we obtain a low degree relation. For instance if $S(x) = x^3 + a\,x^2$, where $a \in \mathbb{F}_{2^n}$ and $c \in \mathbb{F}_{2^n}$ the ciphertext, we have the algebraic system of degree 2:

$$[X_0X_1 + aX_1h - (c^3 + a\,c^2)\,h^2, X_0^2 - X_1h, X_1^2 - X_2h]$$

We compute a Gröbner basis of this system and we find the following equation in degree $m^- = 3$:

$$h^2\left(\left(c^3 + ac^2\right)X_0 + a^2X_1 - X_2 - ahc^2\,(c + a)\right)$$

Note that for this simple example we have in fact a linear relation. In general, we have to compute a Gröbner basis of an algebraic system of degree 2 similarly to equations (1):

$$S^{2^j}(X_0, X_1, \ldots, h) - S(c)^{2^j}\,h^2, X_j^2 - X_{j+1}\,h, \quad j = 0, 1, \ldots \tag{2}$$

Hence we have a very efficient method (the Gröbner basis computation takes only several seconds) to predict the exact behavior of the Gröbner basis computation for an explicit instance of HFE. This could be used to detect weak secret key

Degree D	m	m_{F_5}	m^-
D=3 or D=4	3	3	3
$5 \leq D \leq 12$	4	3	3
D=16	5	3	3
$17 \leq D \leq 96$	5	4	4
D=128	6	4	4
$129 \leq D \leq 512$	6	5	5
$513 \leq D \leq 1024$	6	6	5
$1025 \leq D \leq 1280$	6	6	6

Table 4. Comparison of various bounds m

degree of $S(x)$	$d \leq 16$	$17 \leq d \leq 128$	$129 \leq d \leq 512$
Gröbner complexity	$\mathcal{O}(n^6)$	$\mathcal{O}(n^8)$	$\mathcal{O}(n^{10})$

Table 5. Experimental complexity of the Gröbner basis attack.

instances. We see that when D is big enough $m = m_{F_5} = m^-$. For practical value of D we can establish precisely the experimental complexity of the Gröbner attack (see [11] for details) as follows:

5.2 With fast linear algebra techniques

In this section, we show how fast sparse linear algebra techniques can be used to improve the asymptotic complexity of our attacks. Thanks to the analysis from section 4 and the implied bounds on m, we know that we can find linear relations among all linear combinations of the original multivariate polynomials multiplied by all monomials of degree up to $m-2$. In order to compute these relations, we form a matrix M representing all these polynomials. Each column in the matrix correspond to one of the polynomials, each line is associated to one of the monomials. Each entry is the coefficient in $\mathbb{F}_2$ of the monomial corresponding to the entry's line in the polynomial corresponding to the entry's column. Then, we form a truncated copy M' of M by removing the lines corresponding to the constant and linear monomials. Then, any vector $\boldsymbol{k}$ in the kernel of M', yields a "linear" polynomial $L_{\boldsymbol{k}}$ in x. The expression of this polynomial in term of the elementary polynomials and their multiples can be read directly from $\boldsymbol{k}$. The value of $L_{\boldsymbol{k}}$ is obtained by multiplying the matrix M by $\boldsymbol{k}$ and by reading the values of the lines deleted from M'.

Thus, in order to solve the original HFE cryptosystem, it suffices to compute many independent vectors from the kernel of M'. We further remark that M' is relatively sparse, indeed, each line contains as many entries as the initial quadratic equations. With n variables and n polynomials, there are $O(n^2)$ quadratic monomials and $O(n^{m+1})$ columns in M' and the proportion of non-zero entries is small. Thus, the right approach is to use algorithms that take advantage of this sparsity. Over $\mathbb{F}_2$, we can for example use the block Wiedeman or block Lanczos algorithms (see [6, 19]). T heir complexity to find $O(n)$ elements in the kernel of M' in term of n-bit word operations is the product of the total number of non-zero entries in the matrix by its rank. Approximating the rank by the smallest dimension (i.e., by the number of columns $O(n^{m-1})$), and by counting $O(n^2)$ non-zero entry per column, we find a total complexity of $O(n^{m-1}n^{m-1}n^2) = O(n^{2m})$. We can slightly reduce this complexity by removing most of the extra lines in M'. Indeed, since M' has $O(n^m)$ lines and $O(n^{m-1})$ columns, we can form a truncated copy M'' of M' by randomly removing lines in M' until the number of lines left is almost equal to the number of columns. Any vector in the kernel of M' is of course in the kernel of M''. On the other hand, a vector from the kernel of M'' is, with high probability, in the kernel of M'. If we keep 50 more lines than columns, the probability to find a bad vector in the kernel of M'' and not in the kernel of M' is about 2^{-50}. Assuming that the number of non-zero entry decreases linearly during this randomized truncation, the complexity of the cryptanalysis is lowered to $O(n^{2m-1})$ operations of n-bit words.

5.3 Incremental linear algebra and almost "secret key" recovery

In fact, this cryptanalysis can be further improved by taking into account some specific property of M. The first (and probably most important) remark is that when decrypting many times with the same HFE system the matrix M is almost the same. More precisely, in the basic quadratic system of equation derived from HFE, all the non-constant terms are identical from one resolution to the next. After multiplication by monomials, we find that the lines of the M that correspond to monomials of total degree m and $m-1$ are left unchanged. As a consequence, the contribution of the columns corresponding to original equations multiplied by a monomial of degree $m-2$ or $m-3$ to the kernel elements is left unchanged. This implies that the linear algebra can be split into two parts. We first find the kernel on the submatrix consisting of the lines and columns described above. This is done once for each HFE public key. Once, the kernel vectors on this submatrix are found, we add back the deleted lines and evaluate their values for each of these vectors. In the second part of the linear algebra, which should be performed for each individual decryption, we solve a much smaller system which involve the previously deleted lines and columns. More precisely, we want, for each kernel element from the first phase, to cancel its contribution to the deleted lines. Since the linear algebra in the second phase involves a much smaller matrix, we have almost recovered an equivalent of the secret key. Indeed, after a first expensive step, we can now decrypt quite efficiently.

Looking further into the specific properties of the matrix M we find that the linear algebra can be improved again. Indeed, assume that we want to find the projection of the kernel of M' on a subset of the column vectors. A possible technique is to form a submatrix with these columns and all the lines which are equal to zero on all the other columns. If the number of lines is greater than the number of columns, we probably find the right result. If the number of lines is too small, we find a larger (probably useless) subspace than the kernel. Luckily for us, we can find subsets of the column vectors with enough corresponding lines. One possible construction is to choose a subset S_k of k variables among $x_1, \ldots x_n$ and form the subset of the columns corresponding to the original equations multiplied by monomials of degree $m-2$ formed of the product of $m-2$ variables from S_k. Clearly, the lines corresponding to the product of m variables from S_k are zero outside of the chosen subset. The number of chosen columns is $n \cdot \binom{m-2}{k}$, while the number of lines is $\binom{m}{k}$. From an asymptotic point of view, k is thus of the order of $\sqrt{n}$. This reduces the dimension of the linear system to $O(k^m)$, moreover, the number of non-zero entries per line is now $O(k^2)$. Thus the complexity of the initial step of the attack to $O(k^{2m+2})$ or $O(n^{m+1})$ operations on n-bit words. However, we now need to add the remaining columns and find their contribution. In all generality, it is cumbersome to do this efficiently. Yet, for concrete examples, this is easily done (see section 5.4).

Implementation considerations When implementing the above approach with sparse linear algebra technique, the memory complexity can also be greatly reduced. Indeed, the matrices we are considering have a very regular structure and can be generated by adding together copies of the much smaller matrix that encodes the original public equation. In the copied matrices, the coefficient are moved around through multiplication by some monomial. In sparse linear algebra algorithms, it is important to efficient compute matrix by vector products (for the matrix and its transpose). To save memory, we do not precompute the expanded matrix, but only store the small one. Then the matrix-vector multiplication can be performed by following the structure of the large matrix. For each multiplier monomial, it suffices to extract a small subvector, to multiply it by the small matrix and to add the result in some subvector of the result. Moreover, the result matrix-vector multiplication algorithm is respectful of the memory cache structure. As a consequence, it runs faster than the ordinary sparse matrix multiplication that uses the expanded matrix.

5.4 Application to Quartz

The Quartz cryptosystem [24] is a signature scheme from the HFE family with variation 1 and 3 applied. Each signature computation requires four HFE inversions. The underlying finite field is $\mathbb{F}_{2^{103}}$ and the degree of the secret polynomial is 129. Variation 1 is applied by deleting 3 public equations, variation 3 adds 4 extra variables. Looking up in table 2 at degree $2^3 * 129 = 1032$, we find that $m = 6$. With incremental linear algebra, the starting point is to multiply the 99 public equations by all monomials of degree 4 in 60 variables. This yields a system of dimension around 48 millions and approximately 1800 entries per lines. The estimated complexity of the initial linear algebra step is 2^{62}. While out of range, this is much lower than the estimated security bound of 2^{80} triple DES announced in [24] and [7].

6 Conclusion

We have presented a very efficient attack on the basic HFE cryptosystem based on Gröbner bases computation. It is not only a theoretical attack with a good complexity but also a very practical method since our implementation was able to break the first HFE challenge (80 bits). The results presented in this paper shed a new light on the security of many schemes in the HFE family. The main result is that when the degree D of the secret polynomial is fixed, the cryptanalysis of an HFE system requires polynomial time in the number of variables. Of course, if D and n are large enough, the cryptanalysis may still be out of practical reach. Yet, this shows the need for a re-evaluation of the security of HFE based cryptosystems.

Acknowledgements We would like to thank the J.-F. Michon and D. Augot for their programs. The first named author is indebted to the LIP6 for its partial support of this work (Alpha DS25).

References

1. B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Innsbruck, 1965.
2. B. Buchberger. An Algorithmical Criterion for the Solvability of Algebraic Systems. *Aequationes Mathematicae*, 4(3):374–383, 1970. (German).
3. B. Buchberger. A Criterion for Detecting Unnecessary Reductions in the Construction of Gröbner Basis. In *Proc. EUROSAM 79*, volume 72 of *Lect. Notes in Comp. Sci.*, pages 3–21. Springer Verlag, 1979.
4. M. Bardet, J.-C. Faugère, and B. Salvy. Complexity of Gröbner basis computation for regular, overdetermined . in preparation, 2003.
5. T. Becker and V. Weispfenning. *Groebner Bases, a Computationnal Approach to Commutative Algebra.* Graduate Texts in Mathematics. Springer-Verlag, 1993.
6. D. Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comp*, 62:333–350, 1994.
7. N. Courtois. The security of Hidden Field Equations (HFE). In *Cryptographers' Track RSA Conference*, volume 2020 of *Lectures Notes in Computer Science*, pages 266–281, 2001.
8. N. Courtois, A. Shamir, J. Patarin, and A. Klimov. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology – Eurocrypt'2000*, volume 1807 of *Lectures Notes in Computer Science*, pages 392–407. Springer Verlag, 2000.
9. D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry.* Springer Verlag, New York, 1998.
10. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1–3):61–88, June 1999.
11. J.-C. Faugère. Algebraic cryptanalysis of HFE using Gröbner bases. Technical Report 4738, INRIA, 2003.
12. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero F5. In T. Mora, editor, *Proceedings of ISSAC*, pages 75–83. ACM Press, July 2002.

13. H. Gilbert and M. Minier. Cryptanalysis of SFLASH. In L. Knudsen, editor, *Advances in Cryptology – Eurocrypt'2002*, volume 2332 of *LNCS*, pages 288–298. Springer, 2002.
14. A. Kipnis and A. Shamir. Cryptanalysis of the oil and vinegar signature scheme. In H. Krawczyk, editor, *Advances in Cryptology – Crypto'98*, volume 1462 of *LNCS*, pages 257–266. Springer Verlag, 1998.
15. A. Kipnis and A. Shamir. Cryptanalysis of the HFE Public Key Cryptosystem by R elinearization. In M. Wiener, editor, *Advances in Cryptology – Crypto'99*, volume 1666 of *LNCS*, pages 19–30. Springer Verlag, 1999.
16. D. Lazard. Gaussian Elimination and Resolution of Systems of Algebraic Equations. In *Proc. EUROCAL 83*, volume 162 of *Lect. Notes in Comp. Sci*, pages 146–157, 1983.
17. F. S. Macaulay. *The algebraic theory of modular systems.*, volume xxxi of *Cambridge Mathematical Library.* Cambridge University Press, 1916.
18. T. Matsumoto and H. Imai. Public quadratic polynomial-tuples for efficient signature-veri fication and message-encryption. In C. Günther, editor, *Advances in Cryptology - EuroCrypt '88*, pages 419–454, Berlin, 1988. Springer-Verlag. Lecture Notes in Computer Science Volume 330.
19. P. L. Montgomery. A block Lanczos algorithm for finding dependencies over GF(2). In L. Guillou and J.-J. Quisquater, editors, *Advances in Cryptology - EuroCrypt '95*, pages 106–120, Berlin, 1995. Springer-Verlag. Lecture Notes in Computer Science Volume 921.
20. J. Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme o f Eurocrypt '88. In D. Coppersmith, editor, *Advances in Cryptology - Crypto '95*, pages 248–261, Berlin, 1995. Springer-Verlag. Lecture Notes in Computer Science Volume 963.
21. J. Patarin. *HFE first challenge*, 1996. http://www.minrank.org/challenge1.txt.
22. J. Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In U. Maurer, editor, *Advances in Cryptology - EuroCrypt '96*, pages 33–48, Berlin, 1996. Springer-Verlag. Lecture Notes in Computer Science Volume 1070.
23. J. Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. Extended version, 1996.
24. J. Patarin, L. Goubin, and N. Courtois. Quartz: An Asymetric Signature Scheme for Short Signatu res on PC, submission to NESSIE, 2000.
25. V. Shoup. *NTL 5.3.1, a Library for doing Number Theory*, 2003. `http://www.shoup.net/ntl`.