Towards Algebraic Cryptanalysis of HFE Challenge 2

Mohamed Saied Emam Mohamed[1], Jintai Ding[2], and Johannes Buchmann[1]

[1] TU Darmstadt, FB Informatik
Hochschulstrasse 10, 64289 Darmstadt, Germany
{mohamed,buchmann}@cdc.informatik.tu-darmstadt.de
[2] Department of Mathematical Sciences, University of Cincinnati,
South China University of Technology
jintai.ding@uc.edu

Abstract. In this paper, we present an experimental analysis of HFE Challenge 2 (144 bit) type systems. We generate scaled versions of the full challenge fixing and guessing some unknowns. We use the MXL_3 algorithm, an efficient algorithm for computing Gröbner basis, to solve these scaled versions. We review the MXL_3 strategy and introduce our experimental results.

1 Introduction

Solving systems of multivariate non-linear polynomial equations is one of the important research problems in cryptography. The problem of solving quadratic systems over finite fields is called the Multivariate Quadratic (MQ) problem. This problem is a well-known NP-hard problem and hard on average. Types of public-key encryption and signature schemes, which are based on the intractability of solving the MQ problem, constitute Multivariate Cryptography.

Hidden field equation (HFE) is a multivariate cryptosestem introduced by Patarin in [9]. In the extended version of [9], Patarin introduced two HFE challenges. The first one is an HFE system with 80 quadratic polynomial equations in 80 variables over $\mathbb{F}_2$. The second challenge consists of 144 quadratic equations, 16 of them are hidden, in 144 variables.

Algebraic cryptanalysis has been proposed in the last few years as an effective cryptanalytic method. The secret information of a cryptosystem could be recovered by solving a system of multivariate polynomial equations which describes such cryptosystem [3, 10, 4]. In [6], Faugère and Joux used a version of F_5 algorithm to break the first challenge. In this paper, we present a cryptanalysis of HFE challenge 2 cryptosystems towards an algebraic attack that breaks the full challenge. For this analysis we the MXL_3 algorithm to solve some scaled versions the HFE challenge 2. We present experiments that show how the MXL_3 strategies can solve efficiently these scaled versions.

The paper is organized as follows. In Section 2 we review the HFE cryptosystems. We describe the MXL_3 algorithm in Section 3. Section 4 describes our attack and our experimental results. Before continuing let us introduce the necessary notation.

1.1 Notation

Let $X := \{x_1, \ldots, x_n\}$ be a set of variables, upon which we impose the following order: $x_1 > x_2 > \ldots > x_n$. (Note the counterintuitive $i < j$ imply $x_i > x_j$.) Let

$$R = \mathbb{F}_2[x_1, \ldots, x_n]/\langle x_1^2 - x_1, ..., x_n^2 - x_n\rangle$$

be the Boolean polynomial ring in X with the terms of R ordered by the graded lexicographical order $<_{glex}$. We represent an element of R by its minimal representative polynomial over $\mathbb{F}_2$ where degree of each term w.r.t any variable is 0 or 1.

We denote by $T_d(x_{j_1}, \ldots, x_{j_s})$ the set of terms of degree d in the variables $x_{j_1}, \ldots, x_{j_s}$, and by T_d all the terms of degree d.

Let $P = \{p_1, \ldots, p_m\}$ be set of polynomials in R. A row echelon form is simply a basis for span(P) with pairwise distinct head terms, (see [7] for definition).

We will denote by $P_{(op)d}$ the subset of all the polynomials of degree $(op)d$ in P, where (op) is any of $\{=, <, >, \leq, \geq\}$. A term ordering on R is a total ordering $<$ on $T(R)$ such that: $1 < t, \forall t \in T(R),\ t \neq 1$ and $\forall s, t_1, t_2 \in T(R)$ with $t_1 < t_2$ then $st_1 < st_2$. There are several term orderings. In this paper we use the graded lexicographical term ordering ($glex$. Let $t_1, t_2 \in T(R)$, $t_1 >_{glex} t_2$ if and only if $\deg(t_1) > \deg(t_2)$ or $\deg(t_1) = \deg(t_2)$ and $t_1 >_{lex} t_2$.

Let $p \in \mathbb{F}_q[x_1, \ldots, x_n]$ and the terms in p is ordered by $\leq$. The leading term of p is defined by $\mathrm{LT}(p) := \max_{\leq} \mathrm{T}(p)$, $T(p)$ the set of terms of p.

2 HFE Cryptosystem

We explain the construction of HFE cryptosystem as follows. As any public key cryptosystem, HFE uses two keys, one is public and the other is private. The private key consists of the following: The map φ which transforms a vector $x = (x_1, \ldots, x_n) \in \mathbb{F}_{2^n}$ to a vector $y = (y_1, \ldots, y_m) \in \mathbb{F}_{2^n}$. The transformation φ is a univariate polynomial of degree d in a variable x over an extension field $\mathbb{F}_{2^n}$. The inverse φ^{-1} of φ is easily evaluated over $\mathbb{F}_{2^n}$ by finding a solution for the equation $\varphi(x) = y$. The map φ is chosen such that it can be expressed as a system of n multivariate quadratic polynomial equations over $\mathbb{F}_2$. In this case each coordinate of $\varphi(x)$ is expressed by a polynomial in $x_1, \ldots, x_n$. HFE hides its secret polynomial using two randomly chosen invertible affine transformations (S, T) from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^n}$. The public key is defined by a system of quadratic equations $P = (p_1, \ldots, p_n)$ over $\mathbb{F}_2$, $P = T \circ \varphi \circ S$.

As any MPKC, the HFE security is based on solving a polynomial system $P(x) = c$, where x is an input plaintext and c is the output ciphertext. An HFE system has two parameters that affect the complexity of solving its system. The first parameter is the number of variables (n) and the other is the degree of its secret polynomial (d). The hardness of solving HFE systems is close to solving random systems when d is very big, say $d > 512$). However, the univariate degree d should be small enough to obtain an efficient HFE cryptosystem in practice. In the extended version of [9], Patarin introduced two HFE challenges with a prize US \$500 for attacking any of them. The HFE challenge 1 has parameters $n = 80, d = 96$ and HFE challenge 2 has parameters $n = 36$ and $d = 4352$ over the finite field $\mathbb{F}_{2^4}$, where 4 of the 36 equations are not given public.

The HFE challenge 2 systems can be converted to systems over $\mathbb{F}_2$. The resulting system consists of 144 equations in 144 variables, while 16 of these equations are hidden. In this case, the HFE challenge 2 systems have a special structure over $\mathbb{F}_2$. Let $P = \{p_1, \ldots, p_{36}\}$ are HFE system in $X_1, \ldots, X_36 \in \mathbb{F}_{2^4}$ with a univariate degree $d = 4352$. We can represent each polynomial $p_i \in P$ into 4 polynomials $q_{i_1}, q_{i_2}, q_{i_3}, q_{i_4}$ in $x_1, \ldots, x_{144}$ over $\mathbb{F}_2$. Also, each X_j is represented by 4 new variables $x_{j_1}, x_{j_2}, x_{j_3}, x_{j_4}$. In this case the constructed system over $\mathbb{F}_2$ has a special structure such that no products (terms) of two variables belongs to the same group. For example, let $X_1 \in \mathbb{F}_{2^4}$ be represented by $x_1, x_2, x_3, x_4 \in \mathbb{F}_2$. Then $x_1x_2, x_1x_3, x_1, x_4, x_2x_3, x_2x_4, x_3x_4$ are not appeared in any polynomial of the constructed system over $\mathbb{F}_2$.

3 MXL$_3$ Algorithm

The MXL$_3$ algorithm is a version of the XL algorithm [2] that based on the variable-based enlargement strategy [8, 7], the mutant strategy [5], and a new sufficient condition for a set of polynomials to be a Gröbner basis [7]. In this section, we briefly explain the MXL$_3$.

Let P be a finite set of polynomials in R. Given a degree bound D, the XL algorithm is simply based on extending the set of polynomials P by multiplying each polynomial in P by all the terms in T such that the resulting polynomials have degree less than or equal to D. Then, by using linear algebra, XL computes $\widetilde{P}$, a row echelon form of the extended set P. Afterwards, XL searches for univariate polynomials in $\widetilde{P}$.

In [5], it was pointed out that during the linear algebra step, certain polynomials of degrees lower than expected appear. These polynomials are called mutants. The mutant strategy aims at distinguishing mutants from the rest of polynomials and to give them a predominant role in the process of solving the system. The MutantXL algorithm [5] is a direct application of the mutant concepts to the XL algorithm. It uses mutants (if any) to enlarge the system at the same degree level before it is going to extend the highest degree polynomials and increment the degree level.

In order to specify the enlargement strategy used by MXL$_3$, we need the following additional notation.

Let $X := \{x_1, \ldots, x_n\}$ be a set of variables ordered as $x_1 > x_2 > \ldots > x_n$. Assume the terms of R have been ordered by the graded lexicographical order $<_{glex}$. By an abuse of notation, we call the elements of R polynomials. The leading variable of $p \in R$, LV(p), is defined according to the order defined on X

as

$$\mathrm{LV}(p) := \max\{x \mid \ x \in \mathrm{LT}(p)\}.$$

Let P be a set of polynomials in R, we define the subset $\mathrm{L}(P, x) \subset P$, the *variable partition*, as $\mathrm{L}(P, x) = \{p \in P \mid \ \mathrm{LV}(p) = x\}$.

We have studied the total system of polynomials that are generated by XL. We have observed that each degree part can be partitioned using the leading variable and construct the so called *variable partitions*. When we enlarge the system from degree d to degree $d+1$, the set of degree d polynomials is divided into subsets based on the leading variable of its polynomials. Since the polynomials are ordered using the graded lexicographical order, then the degree d polynomials are partitioned from up to down by $x_1, x_2, \ldots, x_n$ partitions. Only some of these partitions are not empty. Figure 1 shows the structure of the total

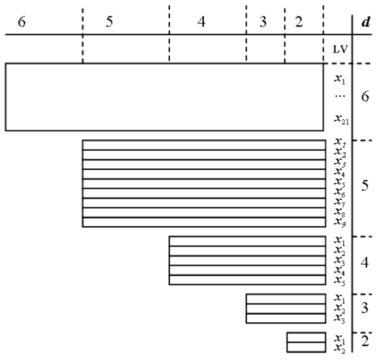


Fig. 1. variable partitions of polynomials generated by XL for a random system of size $n = 26$.

system generated by XL for a random system of size $n = 26$. Horizontal stripes represent non empty variable partitions. For example, at $d = 5$, the degree d polynomials are divided into 9 partitions (x_1-partition,...,x_9-partition). Let the set of polynomials is in the row echelon form, the variable-based enlargement strategy suggests to stepwise constructing the degree 6 polynomials by enlarging one partition per time. In this case, the partition with the smallest leading variable x_9 is enlarged first, then the next smallest x_8, and so on. MXL$_3$ proceeds in this way until it generates lower degree polynomials (mutants) that leads finally to compute a Gröbner basis of the ideal generated by the input set of polynomials. The complete description of MXL$_3$ can be found in [7]

4 Attack Description

In this section, we explain our method to cryptanalysis the second challenge. The HFE challenge 2 can be considered a multivariate digital signature scheme that signs a message of length 128 bits and generates a signature of length 144 bits. It has 36 variables and 32 equations over $\mathbb{F}_{2^4}$. When we transfer the equations over $\mathbb{F}_2$, we have 144 variables and 128 equations. Since we initially construct HFE challenge 2 systems over $\mathbb{F}_{2^4}$, so we select to scale down the parameters of HFE Challenge 2 as follows:

- $\mathbb{F}_{2^4}$: $n = 36$, $h = 4$, and $m = 32 \rightarrow \mathbb{F}_2$: $n = 144$, $h = 16$, and $m = 128$.
- $\mathbb{F}_{2^4}$: $n = 27$, $h = 3$, and $m = 24 \rightarrow \mathbb{F}_2$: $n = 108$, $h = 12$, and $m = 96$.
- $\mathbb{F}_{2^4}$: $n = 18$, $h = 2$, and $m = 16 \rightarrow \mathbb{F}_2$: $n = 72$, $h = 8$, and $m = 64$.
- $\mathbb{F}_{2^4}$: $n = 9$, $h = 1$, and $m = 8 \rightarrow \mathbb{F}_2$: $n = 36$, $h = 4$, and $m = 32$.

We can analysis these systems by applying the following steps on each one of the above systems:

1. Generate a HFE system of equations over $\mathbb{F}_{2^4}$.
2. Remove h equations from the system.
3. Convert the system of equations to be over $\mathbb{F}_2$.
4. Fix the first h variables $(x_1, \ldots, x_h)$.
5. Guess more g variables.
6. Solve the resulting system with size $(n - h - g) \times m$.
7. Repeat the previous two steps with $g = g - 4$ until we reach to g such that the system of size $(n - h - g) \times m$ could not be able to solve.

After converting the system to be over $\mathbb{F}_2$, we fix $n - m$ variables to get a determinant system. After that we guess a number of variables as many as enough for solving the resulting over determined systems easily. We decrease the number of guessing variables by 4 and repeat the previous step until we can not solve the resulting system. For the six step we use our MXL$_3$ implementation to solve the systems. By this way we can estimate the complexity of solving the HFE Challenge 2 systems. In the next section we will present our experimental results and give more analysis.

5 Experimental results

We built our experiments to explain the performance of MXL$_3$ for solving some HFE challenge 2 systems. We run all the experiments on a Sun X4440 server, with four "Quad-Core AMD Opteron$^{\text{TM}}$ Processor 8356" CPUs and 128 GB of main memory. Each CPU is running at 2.3 GHz. We used only one out of the 16 cores.

Table 1 shows results of the HFE challenge 2 system with $n = 144$, $m = 128$, and $h = 16$. We used the method explained in the previous section. After fixing 16 variables we have a system of $m = 128$ equations and variables. We guess more g variables. As the system is originally built over $\mathbb{F}_{2^4}$, then each sequential 4 variables $x_{4i-3}, x_{4i-2}, x_{4i-1}, x_{4i}$ $(i \in \{1, 2, \ldots, m/4\})$ are related since they represent x_i over $\mathbb{F}_{2^4}$. In this case, we choose g such that $g \mid 4$. Moreover, we select the first $g/4$ groups, for example when $g = 40$, we pass values for $x_1, \ldots, x_{40}$.

g	n'	max. matrix	D	Var	Time	Memory
88	40	2600 × 5781	3	x_9	3	3.8
84	44	6444 × 10871	3	x_5	12	13.2
80	48	3668 × 14421	3	x_5	16	24.8
76	52	8804 × 23479	3	x_1	100	61.5
72	56	23452 × 34162	4	x_{37}	272	136
68	60	24692 × 127441	4	x_{21}	14031	1855
64	64	42964 × 238325	4	x_{17}	39547	4819
60	68	196174 × 419753	4	x_{13}	44037	9817
56	72	54772 × 549904	4	x_{13}	144173	19131
52	76	286620 × 887612	4	x_9	365801	47366

Table 1. results of MXL$_3$ for HFE Challenge 2 system ($n = 144$, $m = 128$ and $h = 16$)

Table 2 shows the results of solving some scaled versions of a HFE challenge 2 system with $n = 144$, $m = 128$, and $h = 16$ using Magma's implementation of F$_4$. Magma can not solve any bigger system greater than 128 equations in 72 variables. This explains how our improved MXL$_3$ algorithm is efficient than Magma's F$_4$ in terms of memory. However, F$_4$ is faster than our MXL$_3$ implementation since it uses the advanced Magma's linear algebra techniques.

g	n'	D	Time	Memory
76	52	3	6	203
68	60	4	983	12288
64	64	4	8117	38912
60	68	4	12482	60416
56	72	4	73515	105472
52	76	ran out of memory		

Table 2. results of F$_4$ for HFE Challenge 2 system ($n = 144$, $m = 128$ and $h = 16$)

Table 3 shows how MXL$_3$ solves a scaled version of HFE2 with $n = 72$, $m = 64$, and $h = 8$. Let we fix 8 variables and guess more 8 variables, so the resulting system is 64 equations in 56 variables. As we show from the table at degree $D = 4$, we have nine rounds. Four of them come by enlarging degree 3 partitions of leading variables x_1, x_5, x_9, x_{13} that are generated from the original degree 2 partitions x_1, x_5. The other three partitions $\{x_2, x_3, x_6\}$ are generated by reduction as shown in steps 5, 8. Also, in this level we found few mutants which are not sufficient to solve the system. At $D = 5$, we found some lower degree polynomials generated by reduction in rounds 2,3,4, and 5. While, at round 6 we found a lot of mutants of degree 3 and 4 that successfully solve the system with maximum matrix size 186804×494887.

Step	D	Round	Matrix Size	Rank	Svar	M	UM	MD
1	2	1	64×1597	64	x_1	0	0	-
2	3	1	688×23697	688	x_5	0	0	-
3	3	2	3612×29317	3612	x_1	0	0	-
4	4	1	7484×165068	7484	x_{13}	0	0	-
5	4	2	18132×223897	17780	x_9	1276	232	3
6	4	3	28916×223897	28916	x_9	0	0	-
7	4	4	51182×279217	51000	x_6	0	0	-
8	4	5	105942×300042	83762	x_5	36	24	3
9	4	6	85010×300042	84542	x_5	0	0	-
10	4	7	87230×345568	86582	x_3	0	0	-
11	4	8	161564×370372	135086	x_2	0	0	-
12	4	9	221456×396607	161320	x_1	0	0	-
13	5	1	161384×400975	161368	x_{41}	0	0	-
14	5	2	162332×412111	162256	x_{37}	152	0	4
15	5	3	163228×430256	163228	x_{34}	180	0	4
16	5	4	170040×439111	169820	x_{33}	2120	0	4
17	5	5	172352×477337	172352	x_{30}	480	0	4
18	5	6	186804×494887	186304	x_{29}	4344, 376	376	4, 3

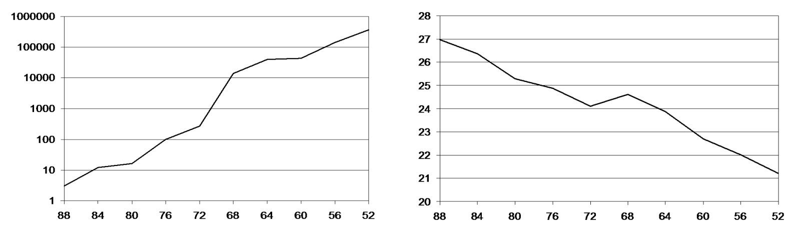
Table 3. Results for HFE2 system $(n = 72, m = 64, h = 8, g = 8)$ by MXL_3

Figure 2 displays the experimental time complexity of solving scaled versions of HFE Challenge 2 system by MXL$_3$ as in Table 1. In this case, after fixing 16 variables (the number of removed equations) we have a HFE system with 128 equations and 128 variables. We guess more g variables and solve the resulting systems with 128 equations and $(128 - g)$ variables.

In Figure 2(a), X-axis represents the number of guessing variables g and Y-axis represents the time consuming to solve each system after guessing g variables. As we show, the time complexity increased as the number of guessing variables decreased. However, this does not give us a real feeling about the complexity of breaking the Challenge.

Figure 2(b) shows the complexity of breaking HFE Challenge 2 in the worst case after guessing different g variables. Here X-axis as in Figure 2(a) represents g, while the values of Y-axis represent the logarithm of the time consuming to solve the scaled system with 128 equations and $128-g$ variables multiplied by 2^g. For example, in the worst case we need 10^{27} seconds to break HFE Challenge 2 when $g = 88$ and around 10^{21} seconds when $g = 52$. It is clear from Figure 2(b) that the time complexity for breaking HFE Challenge 2, in the worst case, decreased as the number of guessing variables decreased.

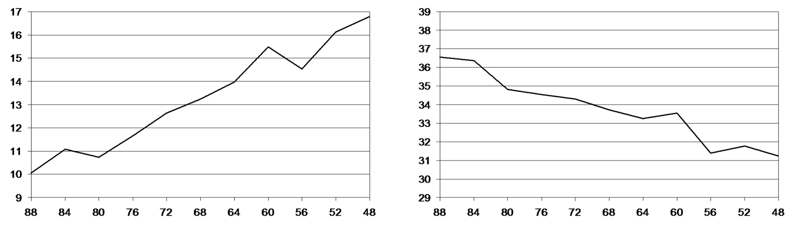
Another study to the complexity of solving HFE Challenge 2 is showed in Figure 3. Since the most time consuming part of MXL$_3$ is the linear algebra step, we study the complexity of computing the row echelon form of the maximum matrix computed by MXL$_3$. Our implementation of MXL$_3$ uses the "Method of Four Russians" [1] in the linear algebra step. The complexity of this method is



(a) (b)

Fig. 2. Explain time complexity in seconds (y-axis) and the number of guessing variables g (x-axis)

$O(N \cdot M \cdot R/\log N)$ [1] where N,M, and R are the number of rows, the number of columns, and the rank respectively. In Figure 3(a), we compute the O-notation for the maximum matrix computed by MXL_3 as in Table 1. In Figure 3(b), we multiply this O-notation by 2^g when we guess g variables. In both figures, Y-axis represents the logarithm of the computed O-notation. Figures 3(a), 3(b) confirm the results that we showed in Figures 2(a), 2(b) respectively. The complexity of computing the row echelon form of the maximum matrix decreased as the number of guessing variables decreased.



(a) (b)

Fig. 3. Relation between the O-Notation of the maximum matrix (y-axis) and the number of guessing variables g (x-axis)

Finally, we interpolate the results in Table 1 to estimate the memory needed to break the full challenge using MXL_3 algorithm. Figure 4 shows the estimated memory consumed for solving scaled versions of HFE challenge 2 systems. We used Lagrange polynomial interpolation method in this computations. In this case, we need approximately 100000 Giga bytes to break the full version of HFE challenge 2.

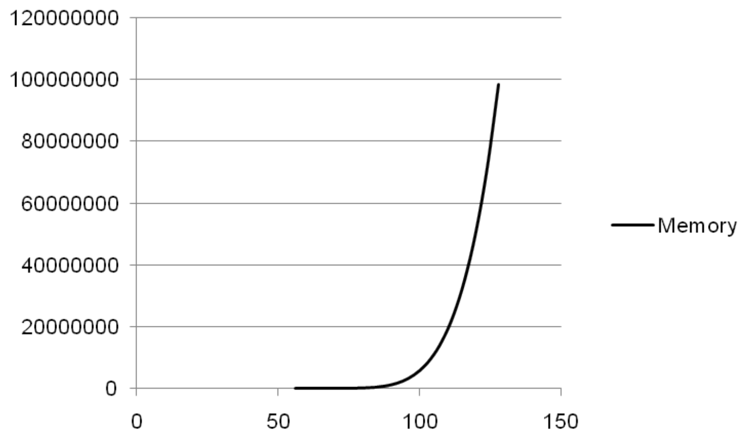


Fig. 4. Relation between the memory usage of solving HFE challenge 2 systems (y-axis) and the number of variables n' (x-axis) after guessing g variables, while the number of equation is fixed ($m = 128$).

Acknowledgments

We want to thank Prof. Dieter Schmidt for supporting us with his implementation of HFE systems.

References

1. G. V. Bard. Accelerating cryptanalysis with the Method of Four Russians. Report 251, Cryptology ePrint Archive, 2006.
2. N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques(EUROCRYPT)*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407, Bruges, Belgium, May 2000. Springer.
3. N. T. Courtois and J. Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In *Proceedings of 8th International Conference on the Theory and Application of Cryptology and Information Security, (ASIACRYPT)*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287, Queenstown, New Zealand, December 2002. Springer-Verlag, Berlin.
4. N. T. Courtois and J. Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. Technical Report 2002/044, Cryptology ePrint Archive, 2002.
5. J. Ding, J. Buchmann, M. S. E. Mohamed, W. S. A. Moahmed, and R.-P. Weinmann. MutantXL. In *Proceedings of the 1st international conference on Symbolic Computation and Cryptography (SCC08)*, pages 16 – 22, Beijing, China, April 2008. LMIB.

6. J.-C. Faugère and A. Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases. In *Proceedings of the 23rd Annual International Cryptology Conference Advances in Cryptology - CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60, Santa Barbara, California, USA, August 2003. Springer.
7. M. S. E. Mohamed, D. Cabarcas, J. Ding, J. Buchmann, and S. Bulygin. MXL3: An efficient algorithm for computing gröbner bases of zero-dimensional ideals. In *Proceedings of The 12th international Conference on Information Security and Cryptology, (ICISC 2009)*, volume 5984 of *Lecture Notes in Computer Science*, pages 87–100, Seoul, Korea, December 2010. Springer-Verlag, Berlin.
8. M. S. E. Mohamed, W. S. A. E. Mohamed, J. Ding, and J. Buchmann. MXL2: Solving polynomial equations over GF(2) using an improved mutant strategy. In *Proceedings of The Second international Workshop on Post-Quantum Cryptography, (PQCrypto08)*, Lecture Notes in Computer Science, pages 203–215, Cincinnati, USA, October 2008. Springer-Verlag, Berlin.
9. J. Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of Asymmetric Algorithms. In *Proceeding of International Conference on the Theory and Application of Cryptographic Techniques Advances in Cryptology- Eurocrypt*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48, Saragossa, Spain, May 1996. Springer.
10. W. Penzhorn. Algebraic attacks on cipher systems. In *Proceedings of 7th AFRICON Conference in Africa, (AFRICON)*, volume 2, pages 969– 974. IEEE, Septmber 2004.