Breaking An Identity-Based Encryption Scheme based on DHIES

Martin R. Albrecht[1] and Kenneth G. Paterson[2⋆]

[1] INRIA, Paris-Rocquencourt Center, SALSA Project UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France CNRS, UMR 7606, LIP6, F-75005, Paris, France
[2] Information Security Group, Royal Holloway, University of London.

1 Introduction

The search for Identity-Based Encryption (IBE) schemes having efficient algorithms and proven security has resulted in many proposals for schemes. A recent proposal, due to Chen *et al.* [4], has very efficient encryption and decryption and claims to have security based on the security of the DHIES public key encryption scheme. In this paper, we present collusion attacks against the scheme of [4] which recover the Trusted Authority's master secret key. These attacks apply even in the weak IBE security model used in [4] and thus completely invalidate the security claims made in [4, Theorem 3.1], the main security result in that paper. We explain where the security analysis in [4] is deficient.

However, it seems that the authors of [4] were already aware of the possibility of collusion attacks against their scheme and set the scheme parameters with the intention of making such attacks inefficient (see Section 3.1 of [4]). We show that their setting of parameters is also flawed by exhibiting a family of efficient collusion attacks against the scheme which applies for their concrete choice of parameters. In fact, our attacks allow the various attack parameters to be traded off against one another – such as the number of private keys needed (the size of the collusion), and the amount of computation needed. For example, we exhibit a master key recovery attack against the concrete scheme of [4] that requires a collusion of 2^{13} parties, $2^{43.3}$ evaluations of a hash function and 2^{30} field operations.

Our attacks show that the scheme cannot be rescued simply by increasing the parameter sizes, without having a serious effect on the efficiency of the scheme.

The remainder of this paper is organised as follows. In Section 2, we present the IBE scheme of [4]. In Section 3 we present some background on Gröbner bases that we use in our attacks. Finally, Section 4 presents our attacks and discusses their application to the concrete parameters selected in [4].

⋆ This author supported by an EPSRC Leadership Fellowship, EP/H005455/1.

2 The IBE Scheme

Our description of the IBE scheme comes from [4, Section 3], with some minor changes to notation. In fact, we do not need to describe the encryption and decryption algorithms of the scheme, since our attacks do not rely on their details. Moreover, we omit some steps that are not needed to understand our attacks.

Setup: This algorithm has as input a security parameter k and a value n (called ℓ in [4]).

1. Run a pairing parameter generator $\mathcal{G}$ to generate a prime q, two groups G_1, G_2 of order q and a pairing $e : G_1 \times G_1 \to G_2$. Let P generate G_1.
2. Set $msk = (u_0, \ldots, u_{n-1}) \in (\mathbb{Z}_q)^n$ where $u_i \leftarrow_R \mathbb{Z}_q$ for $0 \leq i < n$.
3. Set $mpk = (u_0 P, \ldots, u_{n-1} P) \in (G_1)^n$.
4. Let H_0 be a hash function which takes as inputs elements of $\{0,1\}^*$ and outputs subsets of size t of $\{0, 1, \ldots, n-1\}$. A specific method for defining H_0 in terms of elementary hash functions is given in [4]. However, we note that this method is flawed because it does not guarantee to produce an output set of size exactly t. For the concrete parameters selected in [4] ($n = 512$, $t = 8$), a reduced-size output set is quite likely. We ignore this flaw, assuming that the output set is always of size t.
5. Select a pseudo-random number generator (PRNG) F having elements of $\{0,1\}^*$ as seeds and outputs in $\mathbb{Z}_q$.

The master secret key of the scheme is msk. The master public key of the scheme is mpk together with descriptions of G_1, G_2, e, q, P, H_0 and F.

KeyGen: Given an identity $id \in \{0,1\}^*$, this algorithm proceeds as follows:

1. Produce $\{s_0, \ldots, s_{t-1}\} := H_0(id)$.
2. Use F with seed id to produce vectors $a = (a_0, \ldots, a_{t-1}) \in (\mathbb{Z}_q)^t$ and $b = (b_{00}, \ldots, b_{t-1,t-1}) \in (\mathbb{Z}_q)^{\binom{t+1}{2}}$.
3. Output as the private key for identity id the value:

$$x_{id} = \sum_{0 \leq i < t} a_i u_{s_i} + \sum_{0 \leq i \leq j < t} b_{ij} u_{s_i} u_{s_j} \in \mathbb{Z}_q.$$

Notice that the private key corresponding to the identity id is the evaluation of a quadratic function f_{id} over $\mathbb{Z}_q$ in t on the values $u_0, \ldots, u_{n-1}$ from msk. The specific quadratic function f_{id} is determined by H_0 and F and can be readily calculated using only the string id and public information. This observation forms the basis of all of our attacks, the first of which follows immediately.

2.1 A Trivial Collusion Attack

Consider an attacker who has access to private keys x_{id} for strings id of his choice. Such an attacker is standard in the usual security models for IBE, where the attacker has access to a private key extraction oracle. The attacker makes about n^2 such queries for randomly chosen identities. Each query yields the value x_{id} of a new quadratic function f_{id} in the n unknown values u_i. We set up a polynomial system in which variables x_i take the place of the unkown values u_i in the quadratic polynomial $\sum_{0\le i<t} a_i x_{s_i} + \sum_{0\le i\le j<t} b_{ij} x_{s_i} x_{s_j} - x_{id}$ corresponding to f_{id} and x_{id}. After roughly n^2 queries, the quadratic system so obtained can be linearised – replacing each monomial $x_i x_j$ by a new variable y_{ij} yields a full-rank linear system in $\binom{n+1}{2} + n$ variables. The resulting linear system can be solved in time $\mathcal{O}(n^6)$ field operations using elementary linear algebra to recover the master secret key $(u_0, \ldots, u_{n-1})$.

This yields a polynomial-time-and-space attack (in parameter n) for the scheme presented in [4]. In particular, it shows that the security proof of [4, Theorem 3.1], which is the main security result in the paper, cannot be correct. This is because our adversary can be converted into an IND-sID-CCA adversary that breaks the scheme without breaking the underlying DHIES scheme. Technically, since n as defined in [4] is a constant and not dependent on k, our attacker runs with a constant number of extraction queries and in constant time. For the specific parameters $n = 512$, $t = 8$ given in the paper, our attacker runs in time $\mathcal{O}(2^{54})$, where our basic operation is a field operation over $\mathbb{Z}_q$. We note that this kind of attack is anticipated in [4, Section 3.1], which makes it all the more surprising that Theorem 3.1 is included in the paper.

One way in which the main result of [4] could be rescued would be to make n super-polynomial in the security paramter k. However, this would impact adversely on the efficiency of the scheme. Another way would be to limit the number of key extraction queries allowed by the adversary. This latter fix is suggested in [4, Section 3.1], where it is claimed that the scheme will be secure against collusion attacks so long as the adversary is limited to making $0.1n^2$ key extraction queries (whereas the basic attack above requires roughly n^2 such queries). As we shall see with our more sophisticated attacks below, limiting the adversary to $0.1n^2$ key extraction queries is insufficient to make the scheme secure.

2.2 Where the Proof Fails

It is instructive to examine where the proof of [4, Theorem 3.1] fails. In essence, a hash function $H_1 : \{0,1\}^* \to G_2$ is treated as a random oracle in the proof, whereas in fact this hash function must have an algebraic structure derived from the private key equation defining x_{id}. Thus it cannot be modelled as a random oracle.

3 Gröbner Basics

Consider the polynomial ring $P = \mathbb{Z}_q[x_0, \ldots, x_{n-1}]$ over some prime finite field $\mathbb{Z}_q$, some monomial ordering on elements of P and a set of polynomials $f_0, \ldots, f_{m-1}$. We denote by $\mathrm{LM}(f)$ the largest monomial in f and by $\mathrm{LC}(f)$ the coefficient corresponding to $\mathrm{LM}(f)$. We consider the ideal I spanned by $f_0, \ldots, f_{m-1}$.

Definition 1. *Let $f_0, \ldots, f_{m-1}$ be polynomials in P. Define the set*

$$\langle f_0, \ldots, f_{m-1} \rangle = \left\{ \sum_{i=0}^{m-1} h_i f_i : h_0, \ldots, h_{m-1} \in P \right\}.$$

This set I is an ideal and is called the ideal generated by $f_0, \ldots, f_{m-1}$.

A Gröbner basis G for some ideal I is a basis such that for any leading monomial occuring in the ideal there is an element in the basis which has leading monomial dividing it.

Definition 2 (Gröbner Basis). *Let I be an ideal of $\mathbb{Z}_q[x_0, \ldots, x_{n-1}]$ and fix a monomial ordering. A finite subset $G = \{g_0, \ldots, g_{m-1}\} \subset I$ is said to be a* Gröbner basis *of I if for all $f \in I$ there exists a $g_i \in G$ such that $\mathrm{LM}(g_i) \mid \mathrm{LM}(f)$.*

Definition 3 (Reduced Gröbner Basis [5]). *Let I be an ideal of $\mathbb{Z}_q[x_0, \ldots, x_{n-1}]$ and fix a monomial ordering. A finite subset $G = \{g_0, \ldots, g_{m-1}\} \subset I$ is said to be a* reduced Gröbner basis *of I if*

- *G is a Gröbner basis,*
- *$\mathrm{LC}(g_i) = 1$ for all $g_i \in G$, and*
- *for all $g_i \in G$, no monomial of g_i is divisible by some $\mathrm{LM}(g_j)$ for $g_j \in G - \{g_i\}$.*

The reduced Gröbner basis G of I is unique for a given I. From this, it is easy to see that the reduced Gröbner basis of an ideal $I = \langle f_0, \ldots, f_{m-1} \rangle$ with $(u_0, \ldots, u_{n-1}) \in \mathbb{Z}_q^n$ the unique common root of all $f_0, \ldots, f_{m-1}$ is $(x_0 - u_0, \ldots, x_{n-1} - u_{n-1})$. Consequently, if a system of equations has exactly one solution, computing the Gröbner basis is equivalent to solving the system of equations.

Lazard showed in [8] that computing a Gröbner basis for a homogeneous system of polynomials spanning a zero-dimensional ideal can be reduced to linear algebra.

Definition 4. *For the set of m polynomials $f_0, \ldots, f_{m-1}$ we can define and construct the Macaulay matrix $\mathcal{M}_{d,m}^{acaulay}$ of degree d as follows: list "horizontally" all*

the degree d monomials from smallest to largest sorted by some fixed monomial ordering. The smallest monomial comes last. Multiply each f_i by all monomials $t_{i,j}$ of degree $d - d_i$ where $d_i = \deg(f_i)$. Finally, construct the coefficient matrix for the resulting system:

$$\mathcal{M}_{d,m}^{acaulay} = \begin{array}{c} \\ (t_{0,0}, f_0) \\ (t_{0,1}, f_0) \\ (t_{0,2}, f_0) \\ \vdots \\ (t_{1,0}, f_1) \\ \vdots \\ (t_{m-1,0}, f_{m-1}) \\ (t_{m-1,1}, f_{m-1}) \\ \vdots \end{array} \begin{array}{c} \text{monomials of degree } d \\ \left(\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \right. \end{array}$$

Theorem 1 (Lazard's Theorem [8]). *Let $F = \{f_0, \ldots, f_{m-1}\}$ be set of homogeneous polynomials in P spanning a zero-dimensional ideal. There exists a positive integer D for which Gaussian elimination on all $\mathcal{M}_{d,m}^{acaulay}$ for $1 \leq d \leq D$ computes a Gröbner basis for the ideal $\langle F \rangle$.*

The value D is called the degree of semi-regularity and defined as follows:

Definition 5 (D-regular Sequences [2]). *Let $f_0, \ldots, f_{m-1} \subset \mathbb{Z}_q[x_0, \ldots, x_{n-1}]$ be homogeneous polynomials of degrees $d_0, \ldots, d_{m-1}$ respectively. This sequence is regular of degree D if:*

1. *$\langle f_0, \ldots, f_{m-1} \rangle \neq \mathbb{Z}_q[x_0, \ldots, x_{n-1}]$.*
2. *For all $0 \leq i < m$ and $g \in \mathbb{Z}_q[x_0, \ldots, x_{n-1}]$:*

$$\deg(g \cdot f_i) < D \text{ and } g \cdot f_i \in \langle f_0, \ldots, f_{i-1} \rangle \Rightarrow g \in \langle f_0, \ldots, f_{i-1} \rangle.$$

We call D the degree of semi-regularity of the system.

This notion can be extended to affine polynomials by considering their homogeneous components of highest degree. It is conjectured that random systems of polynomials are *semi-regular* systems, i.e. that their degree of regularity is given by:

Lemma 1 (Degree of Semi-Regularity [2]). *The degree of regularity of a semi-regular sequence $f_0, \ldots, f_{m-1}$ of respective degree $d_0, \ldots, d_{m-1}$ is given by the index of the first non-positive coefficient of:*

$$\sum_{k \geq 0} c_k z^k = \frac{\prod_{i=0}^{m-1}(1 - z^{d_i})}{(1-z)^n}.$$

The F_4 [6] and F_5 [7] algorithms can be seen as applications of Lazard's theorem. They successively construct and reduce matrices until a Gröbner basis is found. Consequently, their complexity can be estimated using the degree D.

Theorem 2 (F_5 Complexity [2]). *The complexity of computing a Gröbner basis of a zero-dimensional system of m polynomials in n variables with the algorithm F_5 is*

$$\mathcal{O}\left(\binom{n+D-1}{D}^{\omega}\right)$$

where D is the "degree of semi-regularity" of the system and $2 \leq \omega < 3$ is the linear algebra constant.

Using Theorem 2 and Lemma 1 we can estimate the complexity of solving a random system of equations in n unknowns and m equations. Below, we give concrete values for the $\log_2$ of the expected complexity of the F_5 algorithm in finite field operations for a variety of cases relevant to this work. The table below assumes $\omega = 3$. We note, that assuming $\omega = 3$ is pessimistic, since if the system is dense then there exist practical faster algorithms for Gaussian elimination with $\omega = \log_2 7$ and if the system is sparse, we expect $\omega = 2 + \epsilon$.

	$m =$					
n	$n+1$	$\lfloor n \log n \rfloor$	$\lfloor n \log n^2 \rfloor$	$\lfloor 0.05 \cdot n^2 \rfloor$	$\lfloor 0.1 \cdot n^2 \rfloor$	$0.5n^2$
1024	–	227	176	105	74	–
512	–	165	118	92	65	–
256	–	102	80	80	57	45
128	364	68	49	68	49	38
64	182	49	32	63	41	32
32	91	33	26	54	39	26
16	46	21	21	$m < n$	30	21
8	24	15	11	$m < n$	$m < n$	15

Table 1. Complexity of solving a random system of m equations in n unknowns.

The systems considered in this work are not random since we have the guarantee that only a small fraction of the possible variables appear in a polynomial, that is the systems are sparse. However, we expect that the complexity bounds from Table 1 still apply. In particular, we expect sparse systems to be easier to solve than truly random (dense) systems. Moreover, we do not expect the systems to be harder than random systems. Since the coefficients are chosen uniformly at random each new polynomial does indeed add new useful information to the system and is not a simple algebraic combination of other polynomials already in the system.

In order to verify this intuition, we give concrete running times in seconds on a 2.6 Ghz Opteron for small-scale random systems in n unknowns and $m = \lfloor 0.1n^2 \rfloor$

equations and t unknowns per equation in Figure 1. The x-axis is n, the left-hand side y-axis is the $\log_2$ of the running time and the right-hand side y-axis is the $\log_2$ of the speed-up of $t < n$ compared to $t = n$. We used MAGMA's [3] implementation of the F_4 algorithm and implemented the experiment using the Sage mathematics software [9]. We note that while the values for n are too small to observe the asymptotic behaviour, they show that setting $t = c$ for some constant $c < n$ does not impact the complexity negatively.

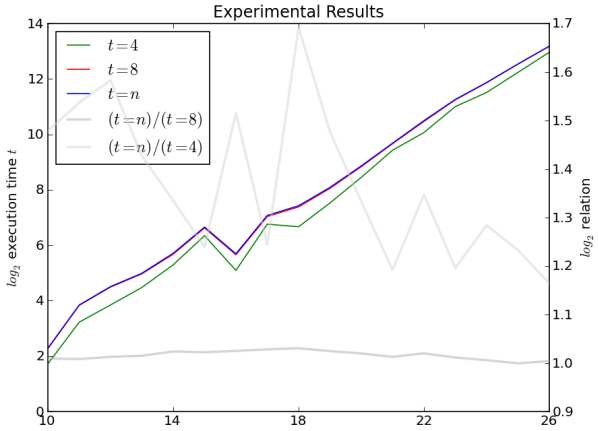

Fig. 1. Experimental verification for $t \in \{4, 8, n\}$ with $m = \lfloor 0.1n^2 \rfloor$.

We note that in Table 1 for $m = n \log n$ the complexity appears to grow sub-exponentially in n, which matches the behaviour over $\mathbb{Z}_2$ for which explicit formulas exist in literature [2]. Thus, we conclude that the statement in [4] that the scheme is secure against collusion attacks involving $m \approx 0.1n^2$ identities is incorrect. In particular, we expect sub-exponential complexity for $m \approx n \log n$.

4 Refining the Basic Attack

We now extend the basic collusion attack shown in Section 2, exploiting the fact that the adversary can precompute the hash function H_0 in an effort to find identities leading to simpler sets of equations in the n unknowns x_i. Our main idea is to partition the set of unknowns into subsets of equal size, and then search for identities yielding systems of equations with unknowns restricted to these sets. Since the subsets are of smaller size than the whole, solving these systems should be easier than solving a single large system.

To this end, we write $n = 2^d$, assuming this parameter is an integer power of 2 (as it is for the concrete parameters in [4]). For some value $s \geq 0$ with $2^{d-s} \geq t$,

we partition the set $\{0, \ldots, n-1\}$ into 2^s sets S_i each containing 2^{d-s} integers by setting $S_i = \{i2^{d-s}, \ldots, (i+1)2^{d-s} - 1\}$. Next we present pseudo-code for our attack.

Set $ID_i := \emptyset, c_i := 0$ for $0 \leq i < 2^s$
Set $c := 0$ and $c_{2^s} := 0$.
repeat
  $c := c + 1$
  $id \leftarrow_R \{0,1\}^*$
  if $H_0(id) \subset S_i$ for some i **then**
    $ID_i = ID_i \cup id$
    $c_i := c_i + 1$
  else
    $c_{2^s} := c_{2^s} + 1$
  end if
until $c_i \geq m$ for $0 \leq i < 2^s$
Obtain x_{id} for the first m elements in each set ID_i using key extraction queries.
for $0 \leq i < 2^s$ **do**
  Using the m values x_{id} for $id \in ID_i$, generate and solve a system of m equations in the set of unknowns $\{x_j : j \in S_i\}$.
end for

The attack involves three main stages: a pre-computation to generate sets of identities ID_i suitable for building systems of equations; making key extraction queries to obtain the required private keys; generating and solving 2^s systems of equations, each containing m equations in 2^{d-s} unknowns. Here s and m are parameters of our attack. It is evident that our attack requires $m2^s$ key extraction queries, and that Table 1 can be used to estimate the running time of the equation solving stage for parameters of interest. We focus next on the running time of the first stage.

The first stage involves making sufficiently many queries to H_0 so as to obtain at least m identities id in each of the 2^s sets ID_i. In practice, the identities id will be drawn from the set of strings of some maximum length. This makes no difference to the attack provided that H_0 acts as a random function from $\{0,1\}^*$ to t-subsets of $\{0, 1, \ldots, n-1\}$. We make this assumption henceforth. Under this assumption, for a uniformly selected id, the probability that $H_0(id) \subset S_i$ is equal to $p := \binom{2^{d-s}}{t} / \binom{n}{t}$.

After c iterations of main loop in the algorithm, the vector $(c_0, \ldots, c_{2^s-1}, c_{2^s})$ follows a multinomial distribution with parameters c and $(p, \ldots, p, 1 - 2^s p)$. It then follows from standard results on the multinomial distribution that, for each $0 \leq i < 2^s$, the random variable c_i follows a binomial distribution with mean cp and variance $cp(1-p)$. Moreover, the covariance between c_i and c_j is equal to $-cp^2$. Since p is generally small for the parameters of interest, the covariance is very small and to a first approximation we may treat the first 2^s values c_i as independent identically distributed random variables. We require each of these

variables to be greater than m for the algorithm to terminate, and we estimate the probability that it does so after no more than c iterations as follows.

Using known estimates for the tail of the binomial distribution (see for example [1, Theorem A.13]), we have, for any real number a:

$$\Pr[c_i \geq cp - a] > 1 - e^{-a^2/2pc}$$

Under our assumption that the c_i may be treated as independent random variables, we then have:

$$\Pr[\bigwedge_{0 \leq i < 2^s} c_i \geq cp - a] > (1 - e^{-a^2/2pc})^{2^s}.$$

Setting $a = m$ and $c = 2m/p$, we then obtain:

$$\Pr[\bigwedge_{0 \leq i < 2^s} c_i \geq m] > (1 - e^{-m/4p})^{2^s} \geq 1 - 2^s e^{-m/4p} \geq 1 - \frac{n}{t} e^{-m/4p}.$$

It is then easy to see that with $p = \binom{2^{d-s}}{t} / \binom{n}{t}$, the algorithm terminates with high probability provided the main loop is executed $c = 2m/p$ times.

A concrete example: We take $n = 512$ (so $d = 9$) and $t = 8$, as in the concrete parameters suggested in [4]. We then select $s = 4$ and $m = 0.5(2^{d-s})^2 = 2^9$ (corresponding to the last column in Table 1). For these parameters, we have $p = 2^{-33.3}$. Then our attack requires $2m/p = 2^{43.3}$ evaluations of H_0 and $m2^s = 2^{13}$ key extraction queries to build $2^s = 16$ sets of 2^9 equations in 32 variables. From Table 1, the complexity of solving the 16 systems of equations is $16 \times 2^{26} = 2^{30}$ field operations. To summarise, we can break the scheme of [4] for the concrete parameters $n = 512$ and $t = 8$ using $2^{43.3}$ evaluations of H_0, 2^{13} key extraction queries and 2^{30} field operations. We remind the reader that our attack recovers the master secret key of the scheme, so our break is a complete one. We also remind the reader that the authors of [4] suggest that their scheme should be secure provided the attacker is limited to making no more than $0.1n^2$ key extraction queries. Here, this equates to $2^{14.68}$ queries, while our attack needs only 2^{13} queries.

A second example (minimising key extractions): We take $n = 512$ (so $d = 9$) and $t = 8$, again as in the concrete parameters suggested in [4]. We then select $s = 6$ and $m = 9$. Then our attack requires $2^{60.8}$ evaluations of H_0, just $2^{9.2}$ key extraction queries and $2^{27.2}$ field operations to recover the master secret key of the scheme.

A third example (doubling the parameters): Suppose the scheme parameters are doubled in an attempt to avoid our attack. So we take $n = 1024$ (so $d = 10$) and $t = 16$. We then select $s = 3$ and $m = 2^{13}$. Our attack then requires $2^{64.1}$ evaluations of H_0, 2^{16} key extraction queries and 2^{41} field operations.

4.1 Refining the Attacks

We have taken a simple approach to generating sets of equations that are easier to solve than sets involving all n variables. The reader can easily see that more sophisticated ways of generating and solving sets of equations can be developed. For example, once the values of some variables are known, we can be less restrictive about how further sets of equations are generated. However, this seems unnecessary in order to conclude that the scheme of [4] is flawed in an essential way.

5 Conclusions

We have presented an analysis of a recently proposed IBE scheme of Chen *et al.* [4]. We have shown that the main security result in [4] does not hold, and that the authors' assessment of their scheme's resistance to collusion attacks is over-optimistic. We have shown how standard Gröbner basis techniques can be used to efficiently break the scheme for the parameters proposed in [4], using a collusion attack with a small number of key extraction queries. Our attack is still practical if all the scheme's parameters are doubled.

We note that in order to achieve 80-bit security against the naive attack from Section 2 one needs at least $n = 2^{14}$. In that case, solving the system of equations would cost $n^6 = 2^{14 \cdot 6} = 2^{84}$ finite field operations. With such parameters the master public key mpk would have a size of $160 \cdot 2^{14}/8 = 320$kb. Note that this variant would still be vulnerable to the attacks in Section 4.

We conclude that the scheme of [4] does not offer an attractive trade-off between security and efficiency.

References

1. N. Alon and J.H. Spencer. The probabilistic method. John Wiley & Sons, 1992.
2. M. Bardet, J. Faugère and B. Salvy. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In *Proceedings of the International Conference on Polynomial System Solving*, pp. 71-74, 2004.
3. W. Bosma, J. Cannon and C. Playoust, The MAGMA Algebra System I: The User Language. In *Journal of Symbolic Computation*, Vol. 24, pp. 235-265. Academic Press, 1997.
4. Y. Chen, M. Charlemagne, Z. Guan, J. Hu and Z. Chen. Identity-based encryption from DHIES. In D. Feng, D.A. Basin and P. Liu *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2010*, pp. 82-88, ACM, 2010.
5. D. Cox, J. Little and D. O'Shea. *Ideals, Varieties, and Algorithms*. 3rd Ed. Springer Verlag. Berlin, Heidelberg, New York, 2007.
6. J.-C. Faugère. A New Efficient Algorithm for Computing Gröbner Bases (F4). In *Journal of Pure and Applied Algebra*, Vol. 139 (1-3), pp. 61-88, 1999.

7. J.-C. Faugère. A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pp.75-83, ACM, 2002.
8. Daniel Lazard. Gröbner-Bases, Gaussian elimination and resolution of systems of algebraic equations. In *Proceedings of the European Computer Algebra Conference on Computer Algebra*, Vol. 162 of LNCS, pp. 146-156, Springer Verlag, Berlin, 1983.
9. W. Stein *et al.* Sage Mathematics Software. Version 4.6.0. 2010