Algebraic Cryptanalysis of the PKC'2009 Algebraic Surface Cryptosystem

Jean-Charles Faugère and Pierre-Jean Spaenlehauer

UPMC, Université Paris 06, LIP6
INRIA Centre Paris-Rocquencourt, SALSA Project
CNRS, UMR 7606, LIP6
Boîte courrier 169 – 4, place Jussieu, 75252 Paris Cedex 05, France
Jean-Charles.Faugere@inria.fr, Pierre-Jean.Spaenlehauer@lip6.fr

Abstract. In this paper, we fully break the Algebraic Surface Cryptosystem (ASC for short) proposed at PKC'2009 [3]. This system is based on an unusual problem in multivariate cryptography: the Section Finding Problem. Given an algebraic surface $X(x, y, t) \in \mathbb{F}_p[x, y, t]$ such that $\deg_{xy} X(x, y, t) = w$, the question is to find a pair of polynomials of degree d, $u_x(t)$ and $u_y(t)$, such that $X(u_x(t), u_y(t), t) = 0$. In ASC, the public key is the surface, and the secret key is the section. This asymmetric encryption scheme enjoys reasonable sizes of the keys: for recommended parameters, the size of the secret key is only 102 bits and the size of the public key is 500 bits. In this paper, we propose a message recovery attack whose complexity is quasi-linear in the size of the secret key. The main idea of this algebraic attack is to decompose ideals deduced from the ciphertext in order to avoid to solve the section finding problem. Experimental results show that we can break the cipher for recommended parameters (the security level is 2^{102}) in 0.05 seconds. Furthermore, the attack still applies even when the secret key is very large (more than 10000 bits). The complexity of the attack is $\widetilde{\mathcal{O}}(w^7 d \log(p))$ which is polynomial with respect to all security parameters. In particular, it is quasi-linear in the size of the secret key which is $(2d + 2)\log(p)$. This result is rather surprising since the algebraic attack is often more efficient than the legal decryption algorithm.

Keywords: Multivariate Cryptography, Algebraic Cryptanalysis, Section Finding Problem (SFP), Gröbner bases, Decomposition of ideals.

1 Introduction

In 1994, Shor designed a quantum algorithm to compute efficiently discrete logarithm and factorization [16]. Hence, if one could construct a quantum computer, a huge number of well established public key cryptosystems – for instance, RSA or Elliptic Curve based systems – would be seriously threatened. Therefore, cryptographers are continuously searching for post-quantum alternatives. The first step to design new cryptosystems is to identify hard problems to use as trapdoors. So far, most of the problems used in post-quantum cryptology can

P.Q. Nguyen and D. Pointcheval (Eds.): PKC 2010, LNCS 6056, pp. 35–52, 2010.

be classified into three main categories: Multivariate cryptography, Code-based cryptography and Lattice-based cryptography.

In this context, Akiyama, Goto, and Miyake propose a new multivariate public-key algorithm at PKC'2009: the Algebraic Surface Cryptosystem (*ASC* for short) [3]. Interestingly, its security is based on a difficult problem which is not common:

Section Finding Problem (SFP). Given an algebraic surface defined by the polynomial $X(x, y, t) \in \mathbb{F}_p[x, y, t]$ (where $\mathbb{F}_p$ denotes the finite field of cardinality p), the question is to find two polynomials $u_x(t), u_y(t) \in \mathbb{F}_p[t]$ of degree d, such that $X(u_x(t), u_y(t), t) = 0$.

As stated in [3], this problem is computationally hard: the only algorithm known so far induces to find roots of a huge multivariate polynomial system. Hence the idea of ASC is to use the surface as public key and the knowledge of a section of this surface as the trapdoor. In comparison to HFE [15] or other multivariate systems, ASC has some interesting and unusual properties. In particular, the keys are unexpectedly short. The security of multivariate systems is usually related to the difficulty of finding a zero of a system of low degree polynomials (often quadratic) in a huge number of variables. For instance, in the case of HFE, the size of the public key is precisely the size of the multivariate system: 265680 bits for a security of 2^{80}. In contrast with HFE, ASC enjoys a small public key of 500 bits for a security of 2^{102}. More generally, for a security level of 2^d, the size of the public key of HFE is $\mathcal{O}(d^3)$. In comparison, the public key of ASC is a unique high degree polynomial in only three variables: its size is $\mathcal{O}(d)$ bits for a security of 2^d. Actually, the authors explains that the keys of ASC are among the shortest of known post-quantum cryptosystems. More precisely, let w denote the degree of the public surface X in x and y. For a security level of p^{2d}, the size of the secret key is $2d\log(p)$ bits and the size of the public key is about $wd\log(p)$. The main observation is that the sizes of the keys are linear in $d\log(p)$, which is the logarithm of the security level.

Although a completely different version of ASC [2] has been attacked by Ivanov and Voloch [11], by Uchiyama and Tokunaga [17] and by Iwami [12], the new version of ASC, presented at PKC'2009, is resistant to all known attacks. We would like to mention that the decryption algorithm raises some questions. Indeed, one step of this algorithm is to recover some factors of given degree D of a univariate polynomial. In order to find those factors, the designers propose to recombine the irreducible factors of the polynomial by solving a knapsack. However, this problem is known to be NP-hard [10]. Therefore, it is not clear if the cryptosystem remains practical for high security parameters.

Main Results. In this paper, we describe a message recovery attack which can break ASC in polynomial time. One important step of the legal decryption algorithm is the factorization of a univariate polynomial. The key idea of the algebraic attack is to perform this factorization step *implicitly* by decomposing ideals deduced from the ciphertext. Indeed, decomposition of ideals can be seen as a generalisation of the standard factorization of polynomials. Hence, this technique allows us to bypass the Section Finding Problem, which is hard.

We present three versions of this attack. The Level 1 Attack is high-level, deterministic, offers a good view of the mechanisms involved and can be implemented straightforwardly into a Computer Algebra System such as Magma (code given in Appendix B). However, this version is not very efficient and cannot break ASC for the recommended parameters. The Level 2 Attack is based on the following observation: the polynomials occurring in ASC have a high degree in t and a rather low degree in x and y. Thus, it is natural to see expressions in t as *coefficients* instead of polynomials in t; in other words, in order to speed up the attack, we have to perform the computations in the ring $\mathbb{F}_p(t)[x, y]$ (where $\mathbb{F}_p(t)$ is the field of fractions) instead of $\mathbb{F}_p[x, y, t]$. In the Level 3 Attack, we replace the ground field $\mathbb{F}_p(t)$ by a finite field $\mathbb{F}_{p^D} \approx \mathbb{F}_p[t]/(P(t))$ for a large enough D to avoid the swelling of the intermediate coefficients and to recover the initial message modulo $P(t)$. Even more efficiently, we can split $P(t)$ into several irreducible factors $P_i(t)$ of small degree; the Chinese Remainder Theorem is then used to recombine the congruences and retrieve the original message. In this third version of the attack, the size of the plaintext determines the number of congruences required as well as the size of the finite fields considered. Therefore, the complexity of the Level 3 Attack is expected to be *quasi-linear* in the size of the secret key. This behaviour is confirmed by experimental results together with a complexity analysis. The binary complexity[1] of the Level 3 Attack is (Theorem 1):

$$\widetilde{\mathcal{O}}(w^6 size(m))$$

where $size(m)$ denotes the binary size of the plaintext, w is the degree of X in the variables x and y and $\widetilde{\mathcal{O}}()$ is the "soft Oh" notation (see e.g. [18, Definition 25.8]). Since the size of the secret key is smaller than $size(m)$, the attack is also quasi-linear in the size of the secret key. In practice, $size(m) \approx dw \log(p)$ (where d is the degree of the secret section). Thus the complexity of the attack is

$$\widetilde{\mathcal{O}}(w^7 d \log(p)).$$

This can be compared with a lower bound on the binary complexity (see page 47) of the decryption algorithm:

$$\widetilde{\mathcal{O}}(\log(p)(w^3 d^3 + dw \log(p))).$$

It can be noted that the decryption algorithm is cubic in the size of the secret key. Therefore, increasing the size of the secret key does not secure the system, since the cost of the decryption algorithm increases faster than the cost of the attack.

We implemented in Magma 2.15-7 the three variants. The Level 3 Attack can break ASC with parameters recommended in [3] ($d = 50$, $p = 2$, $w = 5$) in only 0.05 seconds. Experiments confirm that increasing the size of the secret key with the parameters p and d does not really increase the security of the

[1] The binary complexity is the number of arithmetic operations on bits, whereas the arithmetic complexity is the number of arithmetic operations in the base ring.

system. We are still able to break it in few seconds, even when the size of the secret key is more than 10000 bits! We also try to increase the parameter w (the degree in x and y of the public surface). For a reasonable size of the public key (less than 4000 bits), the message can be recovered in few hours. Finally, we try to figure out whether it is possible to secure the system by increasing the size of the support of the surface (the parameter k). However, as predicted by the complexity analysis, this parameter has very few effect on the complexity of the attack. Thereby, we can consider the system as fully broken.

Structure of the Paper
After this introduction, the paper is organized as follows. In Section 2, we give a short description of the ASC cryptosystem as it is presented in [3]. Then, we explain the theoretical foundations of the attack. In Section 3, we describe the three variants of the attack and we show a concrete example by applying it to the toy example given in [3]. We also perform a precise complexity analysis in Section 5. Finally, we give some experimental results showing that the attack is scalable.

2 Description of the Cryptosystem

We give here a short description of ASC. For a more detailed presentation of this cryptosystem, we refer the reader to [3]. We consider the ring of polynomials $\mathbb{F}_p[x,y,t]$ where p is a prime number. For any polynomial $P \in \mathbb{F}_p[x,y,t]$, Λ_P denotes its support in $\mathbb{F}_p(t)[x,y]$ (that is to say the set of couples $(i,j) \in \mathbb{N}^2$ such that $t^\ell x^i y^j$ is a monomial of P).

2.1 Parameters

The cryptosystem ASC has four parameters: p is the cardinality of the ground field, and d is the degree of the secret section. These two parameters are especially important for the security. They have a direct impact on the binary size of the secret key, which is $2d \log p$. Another parameter is w the degree in x and y of the public surface X. The last parameter is k, the cardinality of Λ_X (which is the support of X in $\mathbb{F}_p(t)[x,y]$). The parameters w, d and p have an impact on the size of the public key which is approximatively $dw \log(p)$ bits.

2.2 Keys

The secret key is a pair of polynomials $(u_x(t), u_y(t))$ of degree d.

The public key is given by:

- A surface described by an irreducible polynomial $X(x,y,t) \in \mathbb{F}_p[x,y,t]$ such that $X(u_x(t), u_y(t), t) = 0$ and $\mathsf{card}(\Lambda_X) = k$.
- Λ_m the support of the plaintext polynomial and $\{d_{ij}^{(m)} \in \mathbb{N}\}_{(i,j)\in\Lambda_m}$ the degrees of the coefficients.
- Λ_f the support of the divisor polynomial and $\{d_{ij}^{(f)} \in \mathbb{N}\}_{(i,j)\in\Lambda_f}$ the degrees of the coefficients.

For encryption/decryption it is required that:

$$\begin{gathered}\Lambda_m \subset \Lambda_f \Lambda_X = \{(i_1+i_2, j_1+j_2) : (i_1, j_1) \in \Lambda_f, (i_2, j_2) \in \Lambda_X\}. \\ \max\{i : (i,j) \in \Lambda_X\} < \max\{i : (i,j) \in \Lambda_m\} < \max\{i : (i,j) \in \Lambda_f\}. \\ \max\{j : (i,j) \in \Lambda_X\} < \max\{j : (i,j) \in \Lambda_m\} < \max\{j : (i,j) \in \Lambda_f\}. \\ \deg_t(X(x,y,t)) < \max\{d_{ij}^{(m)}\}_{(i,j)\in\Lambda_m} < \max\{d_{ij}^{(f)}\}_{(i,j)\in\Lambda_f}.\end{gathered}$$

2.3 Encryption/Decryption

Encryption. Consider a plaintext embedded into a polynomial

$$m(x,y,t) = \sum_{(i,j)\in\Lambda_m} m_{ij}(t) x^i y^j$$

where $\deg(m_{ij}(t)) = d_{ij}^{(m)}$. Choose a random divisor polynomial

$$f(x,y,t) = \sum_{(i,j)\in\Lambda_f} f_{ij}(t) x^i y^j$$

where $\deg(f_{ij}(t)) = d_{ij}^{(f)}$. Then select four random polynomials r_0, r_1, s_0, s_1 such that, for $\ell \in \{0,1\}$,

$$r_\ell(x,y,t) = \sum_{(i,j)\in\Lambda_f} r_{ij}^{(\ell)}(t) x^i y^j, \quad s_\ell(x,y,t) = \sum_{(i,j)\in\Lambda_X} s_{ij}^{(\ell)}(t) x^i y^j$$

and $\forall i,j, \deg(r_{ij}^{(\ell)}(t)) = \deg(f_{ij}(t)), \deg(s_{ij}^{(\ell)}(t)) = \deg(X_{ij}(t))$. Finally, construct the ciphertext $(F_0(x,y,t), F_1(x,y,t))$ where

$$\begin{aligned}F_0(x,y,t) &= m(x,y,t) + f(x,y,t)s_0(x,y,t) + X(x,y,t)r_0(x,y,t), \\ F_1(x,y,t) &= m(x,y,t) + f(x,y,t)s_1(x,y,t) + X(x,y,t)r_1(x,y,t).\end{aligned}$$

Decryption. Consider $h_\ell(t) = F_\ell(u_x(t), u_y(t), t), \ell \in \{0,1\}$ and compute the difference $h_0(t) - h_1(t) = f(u_x(t), u_y(t), t)(s_0(u_x(t), u_y(t), t) - s_1(u_x(t), u_y(t), t))$. Next, find a factor of $h_0(t) - h_1(t)$ whose degree matches $\deg(f(u_x(t), u_y(t), t))$. Let $\widetilde{f}(t)$ denote this factor. Then computes $\widetilde{m}(u_x(t), u_y(t), t) = h_0(t) \mod \widetilde{f}(t)$. Finally, retrieve $\widetilde{m}(x,y,t)$ by solving the linear system:

$$\widetilde{m}(u_x(t), u_y(t), t) = \sum \widetilde{m}_{ijk} u_x(t)^i u_y(t)^j t^k.$$

There are potentially several factors of $h_0(t) - h_1(t)$ whose degree is equal to $\deg(f(u_x(t), u_y(t), t))$. So, we have to verify that we picked the good one. To do so, the designers of ASC propose to use a MAC to verify that $\widetilde{m}(x,y,t) = m(x,y,t)$. If the verification fails, we start again by considering another factor of $h_0(t) - h_1(t)$.

To find factors of $h_0(t) - h_1(t)$ whose degree matches $\deg(f(u_x(t), u_y(t), t))$, the designers propose to factor $h_0(t) - h_1(t)$, then recombine its irreducible factors by solving a knapsack problem. However, the knapsack problem is NP-hard [10]. Therefore, as pointed out in [3], it is not clear if the decryption algorithm remains practicable when the security parameters are high.

2.4 Security of the System

The designers of the cryptosystem propose the following parameters:

- $p = 2$.
- d should be greater than 50.
- $w = \deg_{xy}(X) = \max\{i + j : (i, j) \in \Lambda_X\}$ should be greater than 5.
- The lower bound on k is 3.

The size of the secret key is around 100 bits and the size of the public key is close to 500 bits. According to the designers of ASC, there is so far no known attack faster than exhaustive search for these parameters. Therefore, the security level of ASC is expected to be the cost of exhaustive search of the secret key, namely p^{2d+2}.

3 Description of the Attack

Overview of the Attack

In this section, we propose a message recovery attack on the cryptosystem described above.

The main point of the attack is to decompose ideals, instead of factoring the univariate polynomial obtained by evaluating $F_0 - F_1$ in the section (u_x, u_y). This way, we can implicitly manipulate the so-called *divisor polynomial* f occurring in the decryption process. Consequently, we can avoid to solve the underlying Section Finding Problem, and we obtain a polynomial attack on ASC.

First, we present a high-level and deterministic version of the attack (Algorithm 1) based on two fundamental lemmas. Then, we speed-up the algorithm by considering the field of fractions $\mathbb{F}_p(t)$ (Algorithm 2). Indeed, polynomials occurring in ASC have a high degree in t. Since the complexity of Gröbner bases algorithms is linear in the complexity of the arithmetic in the ground field, it seems natural to compute in the field of fractions $\mathbb{F}_p(t)$. Finally, we use a modular approach to implement efficiently the attack: we perform computations in some well-chosen finite fields $\mathbb{F}_p[t]/(P)$ and recombine the results by using the Chinese Remainder Theorem (Algorithm 3). Doing this, the size of the coefficients of intermediate values are bounded (these coefficients can be huge when computations are performed in the field of fractions). This allows us to break bigger instances of ASC. In particular, we are able to break the system with recommended parameters in 0.05 seconds. Furthermore, this will allow us to perform a precise complexity analysis and to show that this attack is quasi-linear in the size of the secret key. Experimentally, we are able to break with this technique some instances where the size of the secret key is greater than 10000 bits.

Now we compare the efficiency of the three versions of the attack on a small example. For instance, we consider the following parameters $p = 11$, $d = 8$, $w = 5$ and $k = 3$ and we use our Magma implementation. The Level 1 Attack (code given in Appendix) recover the plaintext in 136 seconds. As predicted, the Level 2 Attack is faster and can break the system in 74 seconds. Using the modular approach in the Level 3 Attack really speeds up the computations: it retrieves the plaintext in 0.06 seconds.

3.1 Level 1 Attack: Decomposition of Ideals

The two following lemmas are the key elements of the attack.

Lemma 1. *Let I be the ideal $I = \langle F_0 - F_1, X\rangle \subset \mathbb{F}_p[x, y, t]$. Then $I = I_1 \cap I_2$ where $I_1 = \langle f, X\rangle$ and $I_2 = \langle s_0 - s_1, X\rangle$. Generically, the ideals I_1 and I_2 are prime ideals of $\mathbb{F}_p[x, y, t]$.*

Proof. $I = \langle F_0 - F_1, X\rangle = \langle f\,(s_0 - s_1), X\rangle = I_1 \cap I_2$.

Lemma 1 shows that, once we managed to decompose the ideal $\langle F_0 - F_1, X\rangle = \langle f\,(s_0 - s_1), X\rangle$, we can manipulate implicitly the polynomial f through I_1.

Remark 1. In order to decompose I, a strategy is to eliminate x from I by computing a Gröbner basis of $I \cap \mathbb{F}_p[y, t]$. Generically, this Gröbner basis contains only one polynomial Q. If p is big enough, Q has in general two factors which depend on y and t (we do not consider the factors which are in $\mathbb{F}_p[t]$). This fact is confirmed experimentally. The two factors correspond to I_1 and I_2. Then, we can construct I_1 (resp. I_2) by adding to I an appropriate factor of Q. Since $\deg_y(f) > \deg_y(s_1 - s_0)$, the factor of Q with the highest degree in y is the one corresponding to I_1. To factor efficiently the bivariate polynomial Q, we can use for instance the algorithm in [14].

Lemma 2. *Let J be the ideal of $\mathbb{F}_p[x, y, t]$ generated by $J = \langle F_0, F_1, X\rangle + I_1$. Then $m(x, y, t) \in J$. Moreover, J is a zero-dimensional ideal.*

Proof. $J = \langle F_0, F_1, X\rangle + I_1 = \langle F_0, F_1, X, f\rangle = \langle m, f, X\rangle$.

Remark 2. Lemma 2 shows that we can compute explicitly a multivariate ideal which contains m. Since we know Λ_m, we can recover m by solving the following linear system:

$$NF_J(m) = \sum_{(i,j)\in\Lambda_m} \sum_{k=0}^{d_{ij}^{(m)}} m_{ijk} NF_J(x^i y^j t^k) = 0$$

where NF_J denotes the normal form with respect to the ideal J for a chosen monomial ordering (the definition of the normal form is given in Appendix). Since $\lambda m \in J$ for all $\lambda \in \mathbb{F}_p$, we retrieve m up to multiplication by a scalar.

Remark 3. For efficiency purpose, we compute the Gröbner bases with respect to the *graded reverse lexicographical ordering* (Definition 1 in appendix). Instead of computing the Gröbner basis of $\langle F_0 - F_1, X\rangle \cap \mathbb{F}_p[y, t]$, it is also possible to compute a resultant to eliminate the variable x.

Remark 4. The normal form NF_J is a linear application from $\mathbb{F}_p[x, y, t]$ onto $\mathbb{F}_p[x, y, t]/J$. In the last step of the attack, we are searching for the intersection of its kernel with the $\mathbb{F}_p$-linear subspace generated by Γ_m (where Γ_m denotes the support of m in $\mathbb{F}_p[x, y, t]$). Therefore, the linear system has $\mathsf{card}(\Gamma_m)$ unknowns

Algorithm 1. Level 1 Attack

```
1: Compute a Gröbner basis of the ideal $\langle F_0 - F_1, X\rangle \cap \mathbb{F}_p[y,t]$. Generically this Gröbner basis contains only one polynomial $Q(y,t)$.
2: Factor $Q = \prod Q_i(y,t)$. Let $Q_0(y,t) \in \mathbb{F}_p[y,t]$ denote an irreducible factor with highest degree with respect to $y$.
3: Compute a Gröbner basis of the ideal $J = \langle F_0, F_1, X, Q_0\rangle$.
4: To retrieve the plaintext (up to multiplication by a scalar in $\mathbb{F}_p$), solve the linear system over $\mathbb{F}_p$
       $$\sum_{(i,j)\in\Lambda_m} \sum_{k=0}^{d_{ij}^{(m)}} m_{ijk} NF_J(x^i y^j t^k) = 0.$$
   If the system has no solution, go back to Step 2 and pick another factor of $Q$.
```

and $\deg(J)$ equations ($\deg(J) = \dim(\mathbb{F}_p[x,y,t]/J)$ when $\mathbb{F}_p[x,y,t]/J$ is seen as a $\mathbb{F}_p$-vector space). From the Bézout bound [13], $\deg(J) \approx \deg(m)\deg(X)\deg(f)$. The decryption algorithm requires that $\deg(m(u_x,u_y,t)) \geq \mathsf{card}(\Gamma_m)$ (in order to solve the final linear system) and one can remark that $\deg(X)\deg(f) > \deg(m(u_x,u_y,t)) \approx d\deg_{xy}(m)+\deg_t(m)$ (since $\deg_{xy}(f) > \deg_{xy}(m)$, $\deg_t(f) > \deg_t(m)$ and $\deg(X) > d$). Therefore, the linear system has more equations than unknowns: $\mathsf{card}(\Gamma_m) \leq \deg(m(u_x,u_y,t)) \leq \deg(X)\deg(f) \leq \deg(J)$.

3.2 Level 2 Attack: Computing in the Field of Fractions $\mathbb{F}_p(t)$

Polynomials appearing in ASC have a high total degree, but their degree in the variables x and y is low. Hence, it is natural to consider these polynomials as bivariate polynomials in x and y over the field of fractions $\mathbb{F}_p(t)$. Indeed, the degree in x and y are completely independent of the security parameter d. In this section, we explain how to adapt the attack in this context. Doing that, we expect to have a lower complexity. Indeed, many operations on ideals – for instance Gröbner basis computations – are linear in the complexity of the arithmetic in the ground field.

From now on, $\mathbb{K}$ denotes the field of fractions $\mathbb{F}_p(t)$.

First, we need to transpose the key lemmas in this new context. This can be done for Lemma 1 without any major modification:

Lemma 3. *Let I be the ideal $I = \langle F_0 - F_1, X\rangle$ (seen as an ideal of $\mathbb{K}[x,y]$). Then there exists I_1 and I_2 two strict ideals of $\mathbb{K}[x,y]$ such that $I = I_1 \cap I_2$ and $\langle f, X\rangle \subset I_1$.*

Unfortunately, Lemma 2 cannot be directly transposed in the context of the field of fractions. Indeed, the variety of the ideal $J = \langle F_0, F_1, X\rangle + I_1 = \langle m, f, X\rangle$ (seen as an ideal of $\mathbb{K}[x,y]$) is generically empty since it is generated by three independent equations. Therefore we have to introduce a new variable z if we want to keep the ideal zero-dimensional and strictly included in $\mathbb{K}[x,y,z]$. Roughly speaking, the role of z is to deform the ideal $\langle m, f, X\rangle$ in order to introduce new elements in the variety:

Algorithm 2. Level 2 Attack: computing in the field of fractions $\mathbb{K} = \mathbb{F}_p(t)$

```
1: Compute the resultant $\mathsf{Res}_x(F_0 - F_1, X) \in \mathbb{K}[y]$.
2: Factor the resultant $\mathsf{Res}_x(F_0 - F_1, X) = \prod Q_i(y)$. Let $Q_0(y) \in \mathbb{K}[y]$ denote an irreducible factor of highest degree in $y$.
3: Compute a grevlex-Gröbner basis of the ideal $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset \mathbb{K}[x, y, z]$.
4: Consider the following linear system over $\mathbb{K}$:
     $$NF_J(z) + \sum_{(i,j)\in\Lambda_m} m_{ij}(t) NF_J(x^i y^j) = 0.$$
   If the system has no solution, then go back to Step 2 and choose another factor of the resultant.
5: Return $m = \sum_{(i,j)\in\Lambda_m} m_{ij}(t) x^i y^j$ where $(m_{ij}(t))$ is the unique solution of the linear system.
```

Lemma 4. *Let $J \subset \mathbb{K}[x, y, z]$ be the ideal $J = \langle F_0 + z, F_1 + z, X\rangle + I_1$. Then $m(x, y, t) + z \in J$. Moreover, J is a zero-dimensional ideal.*

Proof. $\langle F_0 + z, F_1 + z, X\rangle + I_1 = \langle F_0 + z, F_1 + z, X, f\rangle = \langle m + z, f, X\rangle$.

To be able to recover the plaintext, we need to solve a linear system with $\mathsf{card}(\Lambda_m)$ unknowns and $\deg(J)$ equations. In practice, there are more equations than unknowns. Thus, if we choose a wrong factor of the resultant (a factor which is not a divisor of f), then the linear system has generically no solution, and we just have to restart from Step 2 until we find an appropriate factor. In practice, the irreducible factor of the resultant with the highest degree in y is almost always a good choice.

Remark 5. It is also possible to combine the two versions of the attack by computing a Gröbner basis of the elimination ideal and factoring it in $\mathbb{F}_p[x, y, t]$, as in Level 1 attack (Steps 1 and 2 in Algorithm 1). Then, once we found $Q_0 \in \mathbb{F}_p[x, y, t]$, we retrieve the message by computing a Gröbner Basis of $J = \langle F_0 + z, F_1 + z, X, Q_0\rangle \subset \mathbb{K}[x, y, z]$ in the field of fractions (Steps $3, 4, 5$ in Algorithm 2).

3.3 Level 3 Attack: Computing in Finite Fields $\mathbb{F}_{p^m}$

In this section, we study how to implement efficiently the attack in practice. In order to speed up the attack and to compute efficiently in the field of fractions, we perform all computations modulo polynomials of $\mathbb{F}_p[t]$. Indeed, a bound on the degree of m with respect to t is known since $\deg_t(m) \leq \max\{d_{i,j}^{(m)}\}$.

We choose a constant C and $n = \deg_t(m)\log(p)/C$ irreducible polynomials $P_1, \ldots, P_n$ of degree close to $C/\log(p)$ such that $\sum \deg(P_i) > \deg_t(m)$. Then for each P_i, we consider

$$\mathbb{F}_p[t]/(P_i) = \mathbb{F}_{p^{\deg(P_i)}}.$$

Algorithm 3. Level 3 Attack: computing in the finite fields $\mathbb{K} = \mathbb{F}_p[t]/(P)$

```
1: Choose $n \approx \deg_t(m)\log(p)/C$ irreducible polynomials of degree $\approx C/\log(p)$ such
   that $\sum \deg(P_i) > \deg_t(m)$.
2: for $i$ from 1 to $n$ do
3:    Consider $\mathbb{K} = \mathbb{F}_p[t]/(P_i)$.
4:    Compute the resultant $\mathsf{Res}_x(F_0 - F_1, X) \in \mathbb{K}[y]$.
5:    Factor the resultant $\mathsf{Res}_x(F_0 - F_1, X) = \prod Q_i(y)$. Let $Q_0(y) \in \mathbb{K}[y]$ denote an
      irreducible factor of highest degree in $y$.
6:    Compute a grevlex-Gröbner basis of the ideal $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset$
      $\mathbb{K}[x, y, z]$.
7:    Consider the following linear system over $\mathbb{K}$:
          $$NF_J(z) + \sum_{(i,j)\in\Lambda_m} m_{ij}(t) NF_J(x^i y^j) = 0.$$
      If the system has no solution, then go back to Step 2 and choose another factor
      of the resultant.
8:    Retrieve a congruence $m \mod P_i = \sum_{(i,j)\in\Lambda_m} m_{ij}(t) x^i y^j$ where $(m_{ij}(t))$ is the
      solution of the linear system.
9: end for
10: Use the CRT to retrieve $m = m \mod \prod P_i$.
```

Considering all computations in $\mathbb{K} = \mathbb{F}_p[t]/(P_i)$ instead of $\mathbb{F}_p(t)$, the attack yields $m \mod P_i$. Finally we use the Chinese Remainder Theorem (CRT) to recover $m \mod \prod P_i$. Since $\deg(\prod P_i) > \deg_t(m)$, we retrieve the plaintext.

Remark 6. The linear system at step 7 in Algorithm 3 has only $\mathsf{card}(\Lambda_m)$ unknowns and $\deg(J) \approx \deg_{xy}(m)\deg_{xy}(f)\deg_{xy}(X)$ equations. For practical parameters, $\mathsf{card}(\Lambda_m) \approx k$ is smaller than $\deg(J)$, thus the linear system is overdetermined and has in general only one solution. This fact is confirmed by experiments.

The value $\sum \deg(P_i) \approx \deg_t(m)$ is only dependent of the size of the plaintext. Therefore, the number of times we have to run the main loop of Algorithm 3 is linear in the size of the plaintext. Since the cost of arithmetic operations in $\mathbb{F}_p[t]/(P)$ only depends on C (which is a constant chosen by the attacker), we expect this Level 3 Attack to be linear or quasi-linear in the size of the plaintext. This expectation will be confirmed by a complexity analysis and by experimental results. Besides, we would also like to mention that the main loop of Algorithm 3 can be easily parallelized.

4 A Concrete Example

We consider here the toy example given in [3]. We have

- $p = 17$.
- The secret key is $(u_x, u_y) = (14t^3 + 12t^2 + 5t + 1, 11t^3 + 3t^2 + 5t + 4)$.

- The public surface is $X = (t+10)x^3y^2 + (16t^2+7t+4)xy^2 + (3t^{16}+8t^{15}+13t^{14}+8t^{13}+3t^{12}+12t^{11}+4t^{10}+8t^9+7t^8+4t^7+13t^6+2t^5+5t^4+4t^3+14t^2+9t+14)$.
- The support of m and f are

$$\Lambda_m = \{(4,4),(0,0)\}, d_{00}^m = 17, d_{44}^m = 17,$$
$$\Lambda_f = \{(5,5),(1,2),(0,0)\}, d_{00}^f = 13, d_{12}^f = 11, d_{55}^f = 18.$$

Here we show how to recover the message m from the ciphertext (F_0, F_1) (given in [3]) with the Level 3 Attack:

1. Since $\deg_t(m) = 17$, we choose (for instance) $P_1, P_2, P_3, P_4 \in \mathbb{F}_p[t]$ irreducible such that $\sum \deg(P_i) \geq 18$. In particular,
$$P_1 = t^5 + t + 14,$$
$$P_2 = t^5 + 14t^4 + 4t^3 + 4t + 4,$$
$$P_3 = t^5 + 9t^4 + 15t^3 + 8t^2 + 4t + 8,$$
$$P_4 = t^5 + 11t^4 + 11t^3 + 8t^2 + 7t + 8.$$
First, we consider the finite field $\mathbb{K} = \mathbb{F}_p[t]/(P_1)$.
2. Compute the resultant in $\mathbb{K}[y]$:
$\mathsf{Res}_x(F_0 - F_1, X) = (9t^4+14t^3+4t^2+6t+13)y^{30} + (5t^4+t^3+14t^2+15t+8)y^{27} + (6t^4+9t^3+10t^2+7t+14)y^{26} + (7t^4+4t^3+8t^2+5t+8)y^{25} + (8t^4+4t^3+7t^2+7t+6)y^{24} + (12t^4+9t^3+8t^2+13t)y^{23} + (9t^4+4t^3+9t^2+15t+6)y^{22} + (3t^4+6t^3+10t^2+6t+6)y^{21} + (9t^4+9t^3+13t^2+15t+6)y^{20} + (4t^4+4t^3+15t^2)y^{19} + (2t^4+11t^3+2t^2+5t+2)y^{16}$.
3. Then factor it in $\mathbb{K}[y]$:
$\mathsf{Res}_x(F_0 - F_1, X) = y^{16}(y+8t^4+3t^3+16t^2+8t+2)(y^2+2t^4+14t^3+14t^2+6t+10)(y^2+15t^4+3t^3+3t^2+11t+7)(y^2+(14t^4+7t^3+4t)y+13t^4+10t^3+7t^2+8t+1)(y^7+(12t^4+7t^3+t^2+5t+15)y^6+(t^4+5t^3+7t^2+12t+11)y^5+(9t^4+14t^3+5t^2+10t+10)y^4+(4t^4+7t^3+t^2+7t+14)y^3+(11t^4+13t^3+12t^2+8t+4)y^2+(15t^4+9t^3+16t^2+14t+14)y+14t^4+3t^3+9t^2+15t+8)$.
4. Consider Q_0 an irreducible factor with highest degree:
$Q_0 = y^7+(12t^4+7t^3+t^2+5t+15)y^6+(t^4+5t^3+7t^2+12t+11)y^5+(9t^4+14t^3+5t^2+10t+10)y^4+(4t^4+7t^3+t^2+7t+14)y^3+(11t^4+13t^3+12t^2+8t+4)y^2+(15t^4+9t^3+16t^2+14t+14)y+(14t^4+3t^3+9t^2+15t+8)$.
5. Compute a Gröbner basis G with respect to the grevlex ordering of the ideal $J = \langle F_0 + z, F_1 + z, X, Q_0 \rangle \subset \mathbb{K}[x,y,z]$.
6. Since $\Lambda_m = \{(0,0),(4,4)\}$ compute $NF_J(x^4y^4)$:
$NF_J(x^4y^4) = N_1 z + N_2 = (15t^4+3t^3+t^2+13t+16)z + (5t^4+11t^2+t+7)$.
7. Solve the linear system $z + m_{44}NF_J(x^4y^4) + m_{00} = 0$ over $\mathbb{K}$:
$$\begin{cases} m_{00} = N_2/N_1 \mod P_1 \\ m_{44} = -1/N_1 \mod P_1. \end{cases}$$
8. Recover a congruence: $m = m_{00} + m_{44}x^4y^4 \mod P_1$.
9. Repeat the process with P_2, P_3 and P_4.

10. Use the CRT to retrieve $m = m \mod \prod P_i$:
$m = (5t^{17}+15t^{16}+4t^{15}+9t^{14}+7t^{13}+2t^{12}+3t^{11}+8t^{10}+11t^{9}+6t^{17}+6t^{8}+3t^{16}+10t^{7}+11t^{15}+7t^{6}+t^{5}+t^{13}+14t^{4}+10t^{12}+3t^{3}+3t^{11}+12t^{2}+8t^{10}+11t+6t^{9}+2)x^4y^4+(13t^8+2t^7+2t^6+10t^5+5t^4+2t^3+15t^2+3t+11)$.

5 Complexity Analysis

In this part, we investigate the complexity of the Level 3 Attack. To simplify the notations, we suppose here that the complexity of multiplying two $n \times n$ matrices is $\mathcal{O}(n^3)$. We note that C is a parameter chosen by the attacker. This parameter fixes the size of the finite fields considered. Indeed, we choose finite fields $\mathbb{K} = \mathbb{F}_p/(P_i)$ with $\deg(P_i) \approx C/\log(p)$. Hence, $\log(\mathsf{card}(\mathbb{K})) \approx C$.

1. First, we estimate the complexity of the computation of the resultant with respect to x in $\mathbb{K}[x,y]$ (where $\mathbb{K} = \mathbb{F}_p[t]/(P_i)$). According to [18] (Corollary 11.18), this can be done in $\widetilde{\mathcal{O}}(w^3)$ operations in $\mathbb{K}$, and the degree of the resultant is $\mathcal{O}(w^2)$.
2. The probabilistic Cantor-Zassenhaus algorithm [18] factors a polynomial of degree n over a finite field $\mathbb{F}_q$ in $\widetilde{\mathcal{O}}(n^2+n\log(q))$ arithmetic operations in $\mathbb{F}_q$. Therefore the arithmetic complexity in $\mathbb{K}$ of the factorization of the resultant is
$$\widetilde{\mathcal{O}}(w^4+w^2\log(\mathsf{card}(\mathbb{K}))) = \widetilde{\mathcal{O}}(w^4+w^2C).$$
3. The degree of regularity of an ideal is an important indicator of the complexity of computing its Gröbner basis with respect to the grevlex ordering: it is the highest degree of the polynomials occuring in the F_5 Algorithm. According to [13,5,4], if an ideal is spanned by m generic equations in n variables, then the complexity of computing a Gröbner basis is:
$$\mathcal{O}\left(m^3\binom{\mathsf{d}_{\mathsf{reg}}+n-1}{n-1}^3\right).$$
Since the ideal $J = \langle m+z, f, X\rangle$ is generated by three independent equations, its degree of regularity can be estimated from the Macaulay bound (see [13]) as
$$\mathsf{d}_{\mathsf{reg}}(J) = (\deg_{xy}(m+z)-1)+(\deg_{xy}(f)-1)+(\deg(X)_{xy}-1)+1.$$
For practical parameters, $\deg_{xy}(m+z) \approx \deg_{xy}(f) \approx \deg(X)_{xy} \approx w$. Therefore, $\mathsf{d}_{\mathsf{reg}} \approx 3w$. The arithmetic complexity in $\mathbb{K}$ of the Gröbner basis computation is then:
$$\mathcal{O}\left(3^3\binom{\mathsf{d}_{\mathsf{reg}}(J)+2}{2}^3\right) = \mathcal{O}(w^6).$$

4. Finally we have a linear system to solve. The number of variables is $\mathsf{card}(\Lambda_m)$. For practical parameters, $\mathsf{card}(\Lambda_m) \approx k$, which is less than 1000 (the recommended parameter is $k = 3$). Hence, this step is negligible in practice compared to the Gröbner basis computation, since an overdetermined linear system with less than 1000 variables in a finite field can be easily solved. Furthermore, this step is analog to the linear system which is solved in the legal decryption algorithm. Therefore this step of the attack is faster than the decryption algorithm which has to be efficient for practical parameters.

The cost of an arithmetic operation in $\mathbb{K}$ is quasi-linear in $\log(\mathsf{card}(\mathbb{K})) \approx C$. The number of times we have to run the main loop of the attack is $size(m)/C$. The complexity of the CRT is $\widetilde{\mathcal{O}}(size(m)\log(size(m)))$ [18]. Putting all the steps together, we find the total complexity of the attack:

Theorem 1. *The total binary complexity of the Level 3 Attack is*

$$\underbrace{\widetilde{\mathcal{O}}(size(m)w^3)}_{resultant} + \underbrace{\widetilde{\mathcal{O}}(size(m)(w^4 + w^2C))}_{factorization} + \underbrace{\widetilde{\mathcal{O}}(size(m)w^6)}_{Gr\ddot{o}bner} + \underbrace{\widetilde{\mathcal{O}}(size(m))}_{CRT}.$$

Hence, the total binary asymptotic complexity of the attack is upper bounded by

$$\widetilde{\mathcal{O}}(w^6 size(m)).$$

Corollary 1. *If we assume that $size(m) \approx wd\log(p)$ (which is the case in practice), then the binary complexity of the attack is: $\widetilde{\mathcal{O}}(dw^7\log(p))$.*

Consequently, the attack is polynomial in all the security parameters and it is quasi-linear in the size of the secret key which is $2d\log(p)$. It can be noted that the parameter k has few effect on the complexity of the attack.

A Lower Bound on the Complexity of the Decryption Algorithm
The complexity of this attack has to be compared with a lower bound on the cost of the decryption process. During the decryption algorithm, one has to factor $(F_0 - F_1)(u_x(t), u_y(t), t)$ over $\mathbb{F}_p[t]$. The degree of this polynomial is at least dw. To the best of our knowledge, the best probabilistic factorization algorithms have an arithmetic complexity of $\widetilde{\mathcal{O}}(d^2w^2 + dw\log(p))$ [18]. Moreover, there is also a knapsack to solve after the factorization. The complexity of this step is difficult to estimate so we do not consider it here (remember that we try to establish a lower bound). The last step of the decryption process is the resolution of a linear system with $\mathcal{O}(dw)$ variables: the arithmetic complexity of this step is $\mathcal{O}(w^3d^3)$. Finally, the total binary complexity of the decryption algorithm is unsharply lower bounded by $\widetilde{\mathcal{O}}(\log(p)(w^3d^3 + dw\log(p)))$ which is cubic in the parameters d and w, and quadratic in $\log(p)$. In comparison, the attack is quasi-linear in d and $\log(p)$, and polynomial of degree 7 in w.

6 Experimental Results

Workstation. The experimental results have been obtained with a Xeon biprocessor 3.2 GHz, with 64 GB of RAM. The instances of ASC have been generated with MAGMA2.15-7. To compute the Gröbner basis, we use the F_4 [7] implementation in MAGMA.

To generate our instances, we pick $\ell, d \in \mathbb{N}$ and we consider the following parameters:

- $w = 2\ell + 5$.
- $\Lambda_m = \{(4+\ell, 4+\ell), (0,0)\}$.
- $\Lambda_X = \{(3+\ell, 2+\ell), (1+\ell, 2+\ell), (0,0)\}$.
- $\Lambda_f = \{(5+\ell, 5+\ell), (1+\ell, 2+\ell), (1,2), (0,0)\}$.
- $\forall (i,j) \in \Lambda_m, d_{ij}^{(m)} = (2\ell+5)d + 21$.
- $\forall (i,j) \in \Lambda_m, d_{ij}^{(f)} = (2\ell+5)d + 22$.

Construction of $\boldsymbol{X}$, $\boldsymbol{u_x}$ and $\boldsymbol{u_y}$

$u_x, u_y \in \mathbb{F}_p[t]$ are random polynomials of degree d.

To construct $X(x,y,t)$, we pick two random polynomials $R_1, R_2 \in \mathbb{F}_p[t]$ of degree 20 and we consider

$$X = R_1(t)(x^{3+\ell}y^{2+\ell} - u_x(t)^{3+\ell}u_y(t)^{2+\ell}) + R_2(t)(x^{1+\ell}y^{2+\ell} - u_x(t)^{1+\ell}u_y(t)^{2+\ell}).$$

Then we verify that $X(x,y,t)$ is irreducible. If not, we restart by picking another R_1 and another R_2.

Table 1 shows the complexity of the Level 3 Attack for different values of p and d. Each entry in the table is obtained by considering the average results over 20 random instances of the cryptosystem.

Table Notations

t_{res} denotes the time used for the computation of the resultant. t_{fact} is the time used by the factorization of the resultant, whereas t_{GB} denotes the cost of the Gröbner basis computation. The time for solving the linear system and for the recombination by the CRT is negligible and hence are not given in the table. According to [3], there were no known attack better than exhaustive search when $d \geq 50$ and $w \geq 5$. Therefore the security bound is the cost of the exhaustive search of the secret section, namely p^{2d+2}.

Interpretation of the Results

It is worth remarking that the first line of Table 1 corresponds to the parameters recommended by the designers [3] and are broken in 0.05 seconds. The major observation is that the complexity of the attack behaves as predicted by the complexity analysis: it is quasi-linear in the parameter d. We also ran some experiments to study the impact of the parameter k (the cardinality of the support of the surface X) on the complexity: as expected, increasing k has very few effect on the cost of the attack. To summarize, we see in Table 1 that trying to secure the system by increasing the size of the secret key (that is to say by increasing the parameters p and d) is pointless: even when the size of the secret key is bigger than 10000 bits, the system can be broken in few seconds.

Table 1. Level 3 Attack – Experimental results with $w = 5$

p	d	w	k	size of public key	size of secret key	t_{res}	t_{fact}	t_{GB}	t_{total}	security bound
2	50	5	3	310 bits	102 bits	0.02s	0.02s	0.01s	0.05s	2^{102}
2	100	5	3	560 bits	202 bits	0.03s	0.02s	0.02s	0.07s	2^{202}
2	200	5	3	1060 bits	402 bits	0.05s	0.05s	0.05s	0.15s	2^{402}
2	400	5	3	2060 bits	802 bits	0.1s	0.1s	0.1s	0.30s	2^{802}
2	800	5	3	4060 bits	1602 bits	0.2s	0.2s	0.2s	0.65s	2^{1602}
2	1600	5	3	8060 bits	3202 bits	0.3s	0.3s	0.4s	1.0s	2^{3202}
2	2000	5	3	10060 bits	4002 bits	0.45s	0.4s	0.4s	1.3s	2^{4002}
2	5000	5	3	25060 bits	10002 bits	0.8s	1.3s	0.8s	3.0s	2^{10002}
17	50	5	3	1267 bits	409 bits	0.2s	2.4s	0.4s	3.0s	2^{409}
17	100	5	3	2289 bits	818 bits	0.3s	5.1s	0.6s	3.0s	2^{818}
17	400	5	3	8420 bits	3270 bits	1.45s	27.7s	3.9s	33.1s	2^{3270}
17	800	5	3	16595 bits	6500 bits	3.1s	70s	9.5s	83s	2^{6500}
10007	500	5	3	34019 bits	13289 bits	29s	217s	64s	310s	2^{13289}

The Parameter *w*

In order to secure the system, one can think of increasing the parameter w since the attack is in $\mathcal{O}(w^7)$. However, we showed that the complexity decryption algorithm is lower bounded by $\mathcal{O}(w^3)$. Consequently, the parameter w should not be too high if the owner of the secret key wants to be able to decrypt. Table 2 gives the experimental results of the attack when w increases.

Interpretation of the Results

The main observation is that the complexity of the attack still behaves as predicted: when w is increased, the Gröbner basis computation is the most expensive step. Increasing w seems to be the best counter-measure against the attack. However, it should be noted that the attack is still feasible in practice, even when the public key is big.

Table 2. Level 3 Attack – Experimental results: increasing w

p	d	w	k	size of public key	size of secret key	t_{res}	t_{fact}	t_{GB}	t_{LinSys}	t_{total}	security bound
2	50	5	3	310 bits	102 bits	0.02s	0.02s	0.01s	0.001s	0.05s	2^{102}
2	50	15	3	810 bits	102 bits	0.7s	0.3s	4.4s	0.03s	5.4s	2^{102}
2	50	25	3	1310 bits	102 bits	3s	1s	32s	0.2s	37s	2^{102}
2	50	35	3	1810 bits	102 bits	10s	3s	260s	1s	274s	2^{102}
2	50	45	3	2310 bits	102 bits	30s	7s	1352s	4s	1393s	2^{102}
2	50	55	3	2810 bits	102 bits	70s	12s	4619s	13s	4714s	2^{102}
2	50	65	3	3310 bits	102 bits	147s	22s	12408s	27s	12604s	2^{102}
2	50	75	3	3810 bits	102 bits	288s	38s	37900s	56s	38280s	2^{102}

7 Conclusion

In this paper, we analyze the security of the PKC'2009 Algebraic Surface Cryptosystem. We provide three variants of a message recovery attack. We also estimate very precisely the complexity of the Level 3 Attack and we show that it is polynomial in all the parameters of the system. Furthermore, it is quasi-linear in the size of the secret key, whereas the decryption algorithm proposed in [3] is cubic.

Experimental results confirm the theoretical analysis. We show that the attack can easily break ASC with recommended parameters. The best choice to try to secure ASC against the attack is to take p and d as small as possible ($p = 2$ and $d = 50$) and increase w. However our implementation is polynomial in w and can break the system in few hours, even when $w = 75$ (this value should be compared to the initial recommended $w = 5$).

Thereby, we consider that the system is fully broken, but we believe that the section finding problem is still an interesting problem; in this paper, we have simply shown how to avoid to solve it in the context of ASC.

Acknowledgements. We wish to thank the anonymous referees for their helpful comments and suggestions. We are also thankful to Maki Iwami for useful discussions.

References

1. Adams, W.W., Loustaunau, P.: An introduction to Gröbner bases. American Mathematical Society (1994)
2. Akiyama, K., Goto, Y.: An Algebraic Surface Public-key Cryptosystem. IEIC Technical Report (Institute of Electronics, Information and Communication Engineers) 104(421), 13–20 (2004)
3. Akiyama, K., Goto, Y., Miyake, H.: An Algebraic Surface Cryptosystem. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, p. 442. Springer, Heidelberg (2009)
4. Bardet, M., Faugere, J.-C., Salvy, B.: 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)
5. Bardet, M., Faugere, J.-C., Salvy, B., Yang, B.-Y.: Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In: Proceedings of the Eight International Symposium on Effective Methods in Algebraic Geometry, MEGA (2005)
6. Cox, D.A., Little, J.B., O'Shea, D.: Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra. Springer, Heidelberg (1997)
7. Faugère, J.-C.: A new efficient algorithm for computing Gröbner bases (F4). Journal of Pure and Applied Algebra 139, 61–88 (1999)
8. Faugère, J.-C.: 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, New York (2002)

9. Faugère, J.-C., Gianni, P., Lazard, D., Mora, T.: Efficient computation of zero-dimensional Gröbner bases by change of ordering. Journal of Symbolic Computation 16(4), 329–344 (1993)
10. Garey, M.R., Johnson, D.S., et al.: Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman, San Francisco (1979)
11. Ivanov, P., Voloch, J.F.: Breaking the Akiyama-Goto cryptosystem. Arithmetic, Geometry, Cryptography and Coding Theory 487 (2009)
12. Iwami, M.: A Reduction Attack on Algebraic Surface Public-Key Cryptosystems. In: Workshop of Research Institute for Mathematical Sciences (RIMS) Kyoto University, New development of research on Computer Algebra, RIMS Kokyuroku, vol. 1572. Springer, Heidelberg (2007)
13. Lazard, D.: Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In: EUROCAL, vol. 162, pp. 146–156. Springer, Heidelberg (1983)
14. Lecerf, G.: New recombination algorithms for bivariate polynomial factorization based on Hensel lifting. To appear in AAECC (2007)
15. Patarin, J.: Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 33–48. Springer, Heidelberg (1996)
16. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: SFCS 1994: Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Washington, DC, USA, pp. 124–134. IEEE Computer Society Press, Los Alamitos (1994)
17. Uchiyama, S., Tokunaga, H.: On the Security of the Algebraic Surface Public-key Cryptosystems. In: Proceedings of SCIS (2007)
18. Von Zur Gathen, J., Gerhard, J.: Modern computer algebra. Cambridge University Press, Cambridge (2003)

A Gröbner Bases and Normal Form

In this section, we shortly describe some fundamental tools from commutative algebra, which are useful for the attack presented in this paper. For a more complete presentation of those tools, the reader can refer to [6,1].

From now on, $\mathbb{K}$ is a field and R denotes the ring $\mathbb{K}[x_1, \ldots, x_n]$. We suppose given an admissible monomial ordering $<$: for the attack we consider the *grevlex* (graded reverse lexicographical) ordering.

Definition 1 (Grevlex ordering). *The* grevlex *ordering is defined as follows. Let $m_1 = x_1^{\alpha_1} \ldots x_n^{\alpha_n}, m_2 = x_1^{\beta_1} \ldots x_n^{\beta_n}$ be two monomials. Then $m_1 > m_2$ if*

- $\sum_{i=1}^{n} \alpha_i > \sum_{i=1}^{n} \beta_i$ *or*
- $\sum_{i=1}^{h} \alpha_i = \sum_{i=1}^{h} \beta_i$ *and the rightmost nonzero entry of* $(\alpha_1 - \beta_1, \ldots, \alpha_n - \beta_n)$ *is negative.*

For any polynomial $P \in R$, $LM(P)$ denotes its leading monomial with respect to $<$. For any ideal $I \subset R$, $LM(I)$ denotes the ideal generated by $\langle \{LM(P) : P \in I\} \rangle$.

Definition 2 (Normal form). *Let I be an ideal of R, and $f \in R$ be a polynomial. Then there exist unique polynomials $h, g \in R$ such that h is monic, $g \in I$, $f = h + g$ and no monomial of h is in $LM(I)$. Then h is called the* normal form *of f with respect to I and $<$, and is noted $NF_I(f)$.*

The normal form is a $\mathbb{K}$-linear application and its main property is:

Proposition 1. *Let I be an ideal of R, and $f \in R$ be a polynomial. Then $f \in I$ if and only if $NF_I(f) = 0$.*

To be able to compute the normal form, we need another fundamental tool: Gröbner bases.

Definition 3 (Gröbner basis). *Let I be an ideal of R. A finite subset of polynomials $G \subset I$ is called a* Gröbner basis *of I (with respect to the monomial ordering $<$) if $\langle LM(G) \rangle = LM(I)$.*

B Magma Code for the Level 1 Attack

In the following piece of code, `p` and `d` are the parameters of the system. `deg_t` is the degree of m with respect to t and `Lambda_m` denotes the support of m (these values are public). `F0` and `F1` are the ciphertext, and `X` is the public surface.

```
R<x,y,t>:=PolynomialRing(GF(p),3,"grevlex");
Res:=Resultant(R!(F0-F1),R!X,x);  // Eliminate x
F:=Factorization(Res); // Factor the resultant
// Pick the irreducible factor of highest degree in y
maxdeg:=Max([Degree(R!f[1],R!y) : f in F]);
exists(Q0){f[1]:f in F| Degree(R!f[1],R!y) eq maxdeg};
J:=Ideal([R!Q0,R!X,R!F0,R!F1]);
Groebner(J);  // Compute the Gröbner basis of J
Coeffm:=PolynomialRing(GF(p),#Lambda_m*(deg_t+1));
R2<x,y,t>:=PolynomialRing(Coeffm,3);
// Construct the linear system
plaintext:=&+[Coeffm.((i-1)*(deg_t+1)+j)*
              R2!NormalForm(R!x^Lambda_m[i][1]*
              R!y^Lambda_m[i][2]*R!t^(j-1),J) :
              i in [1..#Lambda_m], j in [1..deg_t+1]];
// Solve the linear system:
V:=Variety(Ideal(Coefficients(plaintext)));
```