C^*_{-+} and *HM*: Variations Around Two Schemes of T. Matsumoto and H. Imai

Jacques Patarin[1], Louis Goubin[1], and Nicolas Courtois[2]

[1] Bull Smart Cards Terminals
68 route de Versailles - BP 45
78431 Louveciennes Cedex - France
e-mail : {J.Patarin,L.Goubin}@frlv.bull.fr
[2] Modélisation et Signal
Université de Toulon et du Var - BP 132
83957 La Garde Cedex - France
e-mail : courtois@univ-tln.fr

Abstract. In [4], H. Imai and T. Matsumoto presented new candidate trapdoor one-way permutations with a public key given as multivariate polynomials over a finite field. One of them, based on the idea of hiding a monomial field equation, was later presented in [7] under the name C^*. It was broken by J. Patarin in [8]. J. Patarin and L. Goubin then suggested ([9], [10], [11], [12]) some schemes to repair C^*, but with slightly more complex public key or secret key computations. In part I, we study some very simple variations of C^* – such as C^*_{-+} – where the attack of [8] is avoided, and where the very simple secret key computations are kept. We then design some new cryptanalysis that are efficient against some of – but not all – these variations.
[C] is another scheme of [4], very different from C^* (despite the name), and based on the idea of hiding a monomial matrix equation. In part II, we show how to attack it (no cryptanalysis had been published so far). We then study more general schemes, still using the idea of hiding matrix equations, such as *HM*.
An extended version of this paper can be obtained from the authors.

1 Introduction

What is – at the present – the asymmetric signature algorithm with the most simple smartcard implementation (in terms of speed and RAM needed), and not broken ? We think that it is one simple variation of the Matsumoto-Imai C^* algorithm that we present in part I.

C^* was presented in [4] and [7], and was broken in [8], due to unexpected algebraic properties. However, many ways are possible to avoid the cryptanalysis of [8]. In [9], J. Patarin suggested to use a "hidden polynomial" instead of a "hidden monomial". These "HFE" algorithms are still unbroken. However, the secret key computations in HFE schemes are sensibly more complex than in the original C^* scheme. In [10], [11] and [12], J. Patarin and L. Goubin also studied

K. Ohta and D. Pei (Eds.): ASIACRYPT'98, LNCS 1514, pp. 35–50, 1998.

some variations, where the public equations are given in different forms (some of these schemes are also presented in [5]), but here again, in order to avoid the attacks, the secret key computations or the public key computations are generally slightly more complex than in the original C^* scheme.

In part I, we design and study very simple variations of the original C^* scheme. We keep a quadratic public key and the main secret key operation is still the computation of a monomial function $f : x \mapsto x^h$ in a finite field. (The length of the elements of this finite field is much shorter than for RSA, and this explains why the implementations are much more efficient.) We break some of the new variations. However, some others still resist our attacks. They are related to some problems of orthogonal polynomials (how to complete a set of orthogonal polynomials, how to eliminate some random polynomials linearly mixed with orthogonal polynomials, etc).

These variations of C^* can also be applied to the more general HFE scheme of [9] or to Dragon schemes of [10]. We concentrate on C^* because its secret computations are particularly efficient, and because we want to see if these simple ideas can be sufficient or not to enforce the security (in HFE, the analysis is more difficult since no efficient attacks are known at the present).

In part II, we study a very different (despite the name) algorithm of [4], called $[C]$, based on the idea of hiding (with secret affine transformations) a monomial matrix equation. Since the multiplication of matrices is a non-commutative operation, it creates a scheme with very special features. However, as in C^* or HFE, the public key is still given as a set of multivariate polynomials on a finite field, and some of the ideas used in [8] are also useful.

We show how to break the original $[C]$ scheme (no cryptanalysis of this scheme was published before). We then study some more general algorithms, based on the same idea of hiding matrix equations.

Since all those unbroken schemes are new and very similar to broken ones, we certainly do not recommend them for very sensible applications. However, we believe that it is nice to study them because they have very efficient implementations and provide a better understanding of the subtle links between the concept of asymmetric cryptosystem and the computations required for security.

Part I: Variations around C^*

2 A Short Description of HFE and C^*

We present a short description of the HFE and C^* schemes. See [7] (for C^*), or [9] (for HFE) for more details.

The quadratic function f: Let $K = \mathbf{F}_q$ be a finite field of cardinality q. Let $\mathbf{F}_{q^n}$ be an extension of degree n over $\mathbf{F}_q$. Let

$$f(a) = \sum_{i,j} \beta_{i,j} a^{q^{\theta_{ij}} + q^{\varphi_{ij}}} + \sum_k \alpha_k a^{q^{\xi_k}} + \mu \in \mathbf{F}_{q^n}[a]$$

be a polynomial in a over $\mathbf{F}_{q^n}$, of degree d, for integers θ_{ij}, φ_{ij} and $\xi_k \geq 0$.

Since $\mathbf{F}_{q^n}$ is isomorphic to $\mathbf{F}[x]/(g(x))$, if $g(x) \in \mathbf{F}_q[x]$ is irreducible of degree n, elements of $\mathbf{F}_{q^n}$ may be represented as n-uples over $\mathbf{F}_q$, and f may be represented by n polynomials in n variables a_1, ..., a_n over $\mathbf{F}_q$:

$$f(a_1, ..., a_n) = (f_1(a_1, ..., a_n), ..., f_n(a_1, ..., a_n)).$$

The f_i are quadratic polynomials, due to the choice of f and the fact that $a \mapsto a^q$ is a linear transformation of $\mathbf{F}_{q^n}$.

Secret affine transformation of f: Let s and t be two secret affine bijections $(\mathbf{F}_q)^n \to (\mathbf{F}_q)^n$, where $(\mathbf{F}_q)^n$ is seen as an n-dimensional vector space over $\mathbf{F}_q$.

Using the function f above and some representation of $\mathbf{F}_{q^n}$ over $\mathbf{F}_q$, the function $(\mathbf{F}_q)^n \to (\mathbf{F}_q)^n$ that assigns $t(f(s(x)))$ to $x \in (\mathbf{F}_q)^n$ can be written as

$$t(f(s(x_1, ..., x_n))) = (P_1(x_1, ..., x_n), ..., P_n(x_1, ..., x_n)),$$

where the P_i are quadratic polynomials due to the choice of s, t and f.

The "basic" HFE (cf [9]): The public key contains the polynomials P_i, for $i = 1, 2, ..., n$, as above. The secret key is the function f and the two affine bijections s and t as above.
To encrypt the n-uple $x = (x_1, ..., x_n)$, compute the ciphertext $y = (P_1(x_1, .., x_n)$, ..., $P_n(x_1, ..., x_n))$ (x should have redundancy, or a hash of x should also be sent).
To decrypt y, first find all the solutions z to the equation $f(z) = t^{-1}(y)$ by solving a **monovariate** polynomial equation of degree d. This is always feasible when d is not too large (say $d \leq 1000$ for example) or when f has a special shape (as in the case of C^* described below). Next, compute all the $s^{-1}(z)$, and use the redundancy (or the hash of x) to find M from these.
HFE can also be used in signature, as explained in [9] (essentially, the idea is that now x is the signature and y the hash of the message to be signed. If the equation $f(z) = t^{-1}(y)$ has no solution z, we compute another hash).

The C^* algorithm (cf [7]): C^* can be seen as a special case of the more general HFE scheme, where the function f is $f(a) = a^{1+q^\theta}$. Such a function f has some practical advantages: if K is of characteristic 2 and if $1 + q^\theta$ is coprime to $q^n - 1$, then f is a bijection, and the computation of $f^{-1}(b)$ is easy since $f^{-1}(b) = b^{h'}$, where h' is the inverse of $1 + q^\theta$ modulo $q^n - 1$.

However, C^* was broken in [8], essentially because – in the case of a C^* scheme – there always exist equations such as

$$\sum_{i,j} \gamma_{ij} x_i y_j + \sum_i \alpha_i x_i + \sum_j \beta_j y_j + \mu_0 = 0 \tag{1}$$

from which it is possible to break the scheme (see [8]). (Here x is the cleartext (or the signature), y is the ciphertext (or the hash of the message), and γ_{ij}, α_i, β_i and μ_0 are elements of K.) Throughout this paper, we call "equation of type (1)" any equation like (1).

In the case of HFE, no cryptanalysis has yet been found (when f is well chosen), but the secret key computations are more complex.

3 Three Simple Variations of C^* (and HFE)

3.1 Less Public Polynomials: the C^*_- Scheme

The polynomials $(P_1, \ldots, P_n)$ of the "basic" HFE algorithm give y from x. However, it is possible to keep some of these polynomials secret. Let k be the number of these polynomials P_i that we do not give in the public key, so that only P_1, P_2, ..., P_{n-k} are public.

In an encryption scheme, k must be small, because in order to recover x from y, we compute the q^k possibilities for y, compute all the corresponding possible x, and find the good x thanks to the redundancy. When q is not too large, and when k is very small, for example with $k = 1$ or 2, this is clearly feasible.

In a signature scheme, k may be much larger. However, we must still have enough polynomials P_i in order that the problem of finding a value x, whose images by P_1, ..., P_{n-k} are given values, is still intractable. A value $k = 1$, 2, or $k = \frac{n}{2}$ for example may be practical and efficient.

3.2 Introducing Some Random Polynomials: the C^*_+ Scheme

Let P_i be the public polynomials in x_1, x_2, ..., x_n, of a "basic" HFE scheme. We introduce k random extra quadratic polynomials Q_i in x_1, ..., x_n, and we mix the polynomials Q_i and P_i with a secret affine bijection in the given public key.

In a signature scheme, k must be small, because for a given x, the probability to satisfy these extra Q_i equations is $\frac{1}{q^k}$. When m and k are small, the scheme is efficient: after about q^k tries, we obtain a signature.

In an encryption scheme, k may be much larger. However, the total number $k + n$ of quadratic public equations must be such that the problem of finding x from a given y is still intractable (and thus be $< \frac{n(n+1)}{2}$, because with $\frac{n(n+1)}{2}$ equations, the values $x_i x_j$ are found by Gaussian reductions, which gives the x_i). A value $k = 1$, 2 or $k = \frac{n}{2}$ for example may be practical and efficient.

Note: We may combine the variations of sections 3.1 and 3.2. For example, it is possible to design a signature or an encryption scheme from a "basic" HFE with polynomials P_1, ..., P_n, by keeping P_n secret, introducing a random polynomial Q_n instead of P_n, and computing the public key as a secret affine transformation of P_1, ..., P_{n-1}, Q_n. In the case of a C^* scheme, we call C^*_{-+} such algorithms.

3.3 Introducing More x_i Variables

Due to the lack of space, we refer the reader to the extended version of the paper.

4 Toy Simulations of C^*_{-+} with $n = 17$

We have made some toy simulations with $K = \mathbf{F}_2$ and $n = 17$ of C^*_{-+}, with $K = \mathbf{F}_2$ and $n = 17$. (Note that, in real examples, n must be ≥ 64 if $K = \mathbf{F}_2$.)

In all these simulations, we have computed the exact number of independent equations between the 17 bits of the input x_1, ..., x_{17}, and the 17 bits of the output y_1, ..., y_{17} of type (1) (see section 2), or type (2) or (3) defined by:

$$\sum \gamma_{ijk} x_i y_j y_k + \sum \mu_{ij} x_i y_j + \sum \nu_{ij} y_i y_j + \sum \alpha_i x_i + \sum \beta_i y_i + \delta_0 = 0 \quad (2)$$

$$\sum \gamma_{ijk} x_i x_j y_k + \sum \mu_{ij} x_i y_j + \sum \nu_{ij} x_i x_j + \sum \alpha_i x_i + \sum \beta_i y_i + \delta_0 = 0 \quad (3)$$

As shown in table 1, the attacks of [8] do not work directly against C^*_{-+} if we have less public polynomials, at least if $f(x) = x^3$ is avoided and if two or more polynomials are kept secret.

Note: In this table, we have subtracted the number of independent "trivial" equations, such as $x_i^2 = x_i$, or $y_i \cdot$"y_j"$=$"y_i"$\cdot y_j$, where "y_i" and "y_j" are written with their expression in the x_k variables. The notation $[\alpha]$ means that, when the y_k variables are given explicit values, we obtain in average α independent equations in the x_k variables.

Scheme	Type	x^3	x^5	x^9	x^{17}	x^{33}	x^{65}	x^{129}
C^*	(1)	34 [16]	17 [16]	17 [16]	17 [16]	17 [16]	17 [16]	17 [16]
	(2)	612 [16]	340 [16]	323 [16]	340 [17]	323 [16]	374 [16]	323 [16]
	(3)	578 [153]	442 [153]	476 [153]	493 [153]	476 [153]	459 [153]	493 [153]
C^*_{-+1}	(1)	17 [15]	1 [1]	1 [1]	1 [1]	1 [1]	1 [1]	1 [1]
	(2)	340 [15]	52 [15]	36 [15]	36 [15]	36 [15]	87 [15]	36 [15]
	(3)	443 [153]	307 [152]	341 [153]	358 [153]	341 [152]	324 [152]	358 [153]
C^*_{-+2}	(1)	1 [1]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]
	(2)	54 [13]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]
	(3)	309 [151]	173 [135]	207 [151]	224 [152]	207 [150]	190 [152]	224 [153]
C^*_{-+3}	(1)	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]
	(2)	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]
	(3)	176 [153]	51 [68]	74 [91]	91 [108]	74 [91]	57 [74]	91 [108]
C^*_{-+4}	(1)	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]
	(2)	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]	0 [0]
	(3)	44 [61]	0 [17]	0 [17]	0 [17]	0 [17]	0 [17]	0 [17]

Table 1 (for $K = \mathbf{F}_2$ and $n = 17$)

5 First Cryptanalysis of C^*_-

This section in given in appendix 1.

6 The C^*_{--} Algorithm

When $q^r \geq 2^{64}$, the cryptanalysis given in section 5 is not efficient. The scheme is then called C^*_{--}. It cannot be used for encryption any more, but it is still a very efficient scheme for signatures, and its security is an open problem.

7 Cryptanalysis of C^*_+

The cryptanalysis of C^*_+ is very simple: it just works exactly as the original cryptanalysis of C^*. We first generate all the equations of type (1). Since in C^*_+, we just have **added** some equations (and eliminated none), we find at least as much equations (1) as in the original C^*, from which – as explained in [8] – we can find x from y (and thus break the system). Moreover, we can eliminate the random added equations and recover an original C^*, because an equation (1) generally comes from only the y_i of C^* (and not from the added equations). Therefore, by writing (1) as $x_1(P_1(y)) + x_2(P_2(y)) + \ldots + x_n(P_n(y))$ (where P_1, ..., P_n are polynomials of degree one in y_1, ..., y_{n+k}), and by making the change of variables $y'_1 = P_1(y)$, ..., $y'_n = P_n(y)$, the variables y'_1, ..., y'_n are the outputs of an original C^* scheme.

Note: However, this idea of adding an equation may be much more efficient in a scheme where no equation (1) exist (as in some HFE schemes) (or when we add **and** eliminate some equations, as in C^*_{-+} below).

8 Cryptanalysis of C^*_{-+}, Second Cryptanalysis of C^*_-

Cryptanalysis of C^*_{-+1}: We know that from the variables of the original C^* we have at least n independent equations of type (1), so that – by multiplying these equations by one x_k, $1 \leq k \leq n$ – we generate n^2 independent equations of type (3).

By Gaussian reductions, we obtain at least $n^2 - \frac{n(n+1)}{2}$ $\left(= \frac{n(n-1)}{2}\right)$ equations of type (3) with no terms in y_1 (because we have at most $\frac{n(n+1)}{2}$ terms in $y_1 x_i x_j$ or $y_1 x_i$). Giving then explicit values for y, we obtain (by Gaussian reductions on the $X_{ij} = x_i \cdot x_j$ variables) the x_i values. As a result, with the equations (3) we can break C^*_{-+1}, *i.e.* recover an x from a given y. This attack works because (as shown in our simulations of section 4) the number of independent equations does not decrease significantly when the y_k variables are given explicit values.

Cryptanalysis of C^*_{-+r}, for $r = 2, 3$: As shown in table 1, we generally have more than $\frac{n(n-1)}{2}$ equations of type (3), so that the attack also works very well when $r = 2$ or $r = 3$, since we have more equations (3) than expected. Of course, when – after Gaussian reductions – we still have a few variables to guess, we can guess them by exhaustive search (if this number is very small).

Cryptanalysis of C^*_{-+r}, for $r \geq 4$: When $r \geq 4$, the attack given above may not work, so that we may need to generalize this attack by generating more general equations such as equations of total degree $d \geq 4$ (instead of three), and of degree one in the y_i variables.

We know that from the variables of the original C^* we have at least n independent equations of type (1). So by multiplying these equations by $d-2$ variables x_k, $1 \leq k \leq n$, we generate about $n \cdot \frac{n^{d-2}}{(d-2)!}$ independent equations of the following type:

$$\sum \gamma_{i_1 i_2 \ldots i_d} x_{i_1} x_{i_2} \ldots x_{i_{d-1}} y_d + \ldots = 0. \qquad (*)$$

By Gaussian reductions, we obtain at least $n \cdot \frac{n^{d-2}}{(d-2)!} - r \cdot \frac{n^{d-1}}{(d-1)!}$ equations $(*)$ with no terms in y_1, y_2, ..., y_r (because we have at most $\frac{n^{d-1}}{(d-1)!}$ terms in $y_\mu x_{i_1} x_{i_2} \ldots x_{i_{d-1}}$, and r values μ such that $1 \leq \mu \leq r$). Giving then explicit values for y, we obtain (by Gaussian reductions on the $X_{i_1 \ldots i_{d-1}} = x_{i_1} \ldots x_{i_{d-1}}$ variables) the x_i values if $n \cdot \frac{n^{d-2}}{(d-2)!} - r \cdot \frac{n^{d-1}}{(d-1)!} \geq \frac{n^{d-1}}{(d-1)!}$ (because as shown in our simulations the number of independent equations does not dramatically decrease when we give explicit values for y), *i.e.* when $r \leq d - 2$.

Complexity: The complexity of this attack is essentially the complexity of Gaussian reductions on $\mathcal{O}(n^d)$ terms, *i.e.* $\mathcal{O}(n^{\omega d})$, with $\omega = 3$ in the usual Gaussian reduction algorithms, or $\omega = 2.3755$ in the best known general purpose Gaussian reduction algorithm (see [1]). As a result, this complexity increases in $\mathcal{O}(n^{\omega r})$, *i.e.* exponentially in r.

Since our simulations show that this attack works sensibly better than described above (because we have a few more equations $(*)$), we may expect to attack C^*_{-+r} when $r \leq 10$ approximately. Therefore, we think that any $r \leq 10$ is insecure. However, the complexity of the attack increases a lot when r increases. Hence, at the present, for practical applications, it is an open problem to find efficient cryptanalysis of C^*_{-+r} when $r > 10$.

Can we recover the corresponding C^*_- from C^*_{-+} ?
This is sometimes feasible. For example, when we have equations of type (2) (this is generally the case only when r is very small: see table 1), they generally come from y_k variables of the original C^*_-, and not from the added random quadratic equations. Therefore, by looking at the terms in factor of a monomial $x_i x_j$ in those equations (2), we find the vector space generated by the public equations of the original C^*_- equations. (C^*_{-+} can then be attacked as a C^*_- algorithm.)

Part II: Schemes with a Hidden Matrix

9 The [C] Scheme

Let us recall the description of the $[C]$ scheme, presented by H. Imai and T. Matsumoto in [4]. Let $K = \mathbf{F}_{2^m}$ be a *public* finite field of cardinality $q = 2^m$. The basic idea is to use the transformation $A \mapsto A^2$ of the set $\mathcal{M}_2(K)$ of the 2×2 matrices over the field K.

This transformation is not one-to-one, but it can be proved (see the extended version) that its restriction Φ to the set $\mathcal{E} = \{M \in \mathcal{M}_2(K), \mathrm{tr}(M) \neq 0\}$ is a bijection whose inverse is given by:

$$\Phi^{-1}(B) = \frac{1}{\sqrt{\mathrm{tr}(B)}} \cdot \Big(B + \sqrt{\det(B)} \cdot I\Big),$$

where $\sqrt{\ }$ denotes the inverse of the bijection $\lambda \mapsto \lambda^2$ on $\mathbf{F}_{2^m}$. The function $\sqrt{\ }$ is easy to compute, since $\sqrt{\lambda} = \lambda^{2^{m-1}}$ for any $\lambda \in \mathbf{F}_{2^m}$.

The set $\mathcal{M}_2(K)$ can be considered as a vector space of dimension 4 over K. Therefore, we can choose $s : K^4 \to \mathcal{M}_2(K)$ and $t : \mathcal{M}_2(K) \to K^4$ two *secret* linear bijections such that s maps the hyperplane $\{x_1 = 0\}$ of K^4 onto the hyperplane $\{\mathrm{tr}(M) = 0\}$ of $\mathcal{M}_2(K)$, whereas t maps the hyperplane $\{\mathrm{tr}(M) = 0\}$ of $\mathcal{M}_2(K)$ onto the hyperplane $\{x_1 = 0\}$ of K^4.

Each message M is represented by a 4-uple $(x_1, x_2, x_3, x_4) \in K^4$ such that $x_1 \neq 0$. The message space is $\mathcal{M} = \{(x_1, x_2, x_3, x_4) \in K^4,\ x_1 \neq 0\}$.

The quadratic function f is defined on the message space by:

$$f : \begin{cases} \mathcal{M} \to \mathcal{M} \\ x \mapsto t\big(s(x)^2\big) \end{cases}.$$

The hypotheses made on s and t show that the function f is a bijection.

[C] **used in encryption mode:** The public key is the 4-uple (P_1, P_2, P_3, P_4) of 4-variate quadratic polynomials over K that represent f. They are defined by:

$$f(x_1, ..., x_4) = \big(P_1(x_1, ..., x_4), P_2(x_1, ..., x_4), P_3(x_1, ..., x_4), P_4(x_1, ..., x_4)\big).$$

The secret key is the two linear bijections s and t.

To encrypt the message M represented by $x = (x_1, x_2, x_3, x_4) \in \mathcal{M}$, compute the ciphertext $y = (y_1, y_2, y_3, y_4)$ with the following formulas:

$$(\mathcal{S}) \quad \begin{cases} y_1 = P_1(x_1, x_2, x_3, x_4) \\ y_2 = P_2(x_1, x_2, x_3, x_4) \\ y_3 = P_3(x_1, x_2, x_3, x_4) \\ y_4 = P_4(x_1, x_2, x_3, x_4) \end{cases}$$

To decrypt the ciphertext $y \in \mathcal{M}$, compute:

$$x = s^{-1}\left(\frac{1}{\sqrt{\mathrm{tr}(t^{-1}(y))}} \cdot \Big(t^{-1}(y) + \sqrt{\det(t^{-1}(y))} \cdot I\Big)\right).$$

10 First Cryptanalysis of [C]

This section is given in appendix 2.

11 The More General $[C_n]$ Scheme

The $[C_n]$ scheme is a generalization of $[C]$, which involves $n \times n$ matrices over K, instead of 2×2 matrices.

As in the case of $[C]$, we take a public finite field $K = \mathbf{F}_{2^m}$ of cardinality $q = 2^m$. The basic idea is still to use the transformation $A \mapsto A^2$ of the set $\mathcal{M}_n(K)$ of the $n \times n$ matrices over the field K. The set $\mathcal{M}_n(K)$ can be considered as a vector space of dimension n^2 over K, so that we can choose $s : K^{n^2} \to \mathcal{M}_n(K)$ and $t : \mathcal{M}_n(K) \to K^{n^2}$ two *secret* affine bijections.

Each message M is represented by a n^2-uple $(x_1, ..., x_{n^2}) \in K^{n^2}$. The message space is $\mathcal{M} = K^{n^2}$.

The quadratic function f is defined on the message space by:

$$f : \begin{cases} \mathcal{M} \to \mathcal{M} \\ x \mapsto t\big(s(x)^2\big) \end{cases}.$$

$[C_n]$ **used in encryption mode:** The public key is the n^2-uple $(P_1, ..., P_{n^2})$ of n^2-variate quadratic polynomials over K that represent f. They are defined by:

$$f(x_1, ..., x_{n^2}) = \big(P_1(x_1, ..., x_{n^2}), ..., P_{n^2}(x_1, ..., x_{n^2})\big).$$

The secret key is the two affine bijections s and t.

To encrypt the message M represented by $x = (x_1, ..., x_{n^2}) \in \mathcal{M}$, compute the ciphertext $y = (y_1, ..., y_{n^2})$ with the following formulas:

$$\begin{cases} y_1 = P_1(x_1, ..., x_{n^2}) \\ \quad \vdots \\ y_{n^2} = P_{n^2}(x_1, ..., x_{n^2}) \end{cases}$$

To decrypt the ciphertext $y \in \mathcal{M}$, one has to solve the equations $A^2 = B$, where $B = t^{-1}(y)$, and then to compute the cleartext $x = s^{-1}(A)$.

# pre-images	$n = 2$	$n = 3$	$n = 4$	# pre-images	$n = 2$	$n = 3$	$n = 4$
0	6	252	34440	13-15	0	0	0
1	8	160	22272	16	0	0	672
2	0	42	5040	17-21	0	0	0
3	0	0	0	22	0	2	240
4	2	56	2240	23-315	0	0	0
5-11	0	0	0	316	0	0	2
12	0	0	630	> 316	0	0	0

Table 2: number of pre-images for $[C_n]$ over $\mathbf{F}_2$ (toy examples)

It is important to notice that $A \mapsto A^2$ is not a bijection any longer (contrary to the original $[C]$ scheme described in section 9). As a result, there may be several possible cleartexts for a given ciphertext. One solution to avoid this

ambiguousness is to put some redundancy in the representation of the messages, by making use of an error correcting code or a hash function (for details, see [9] p. 34, where a similar idea is used in a different scheme).

The feasibility of choosing the right cleartext among the possible ones is due to the fact that – for an average B – the number of solutions A of the equations $A^2 = B$ remains reasonable, as shown in table 2 above.

To solve the equation $A^2 = B$ when $B \in \mathcal{M}_n(K)$, two methods can be used:

The first one is based on the Jordan reduction of matrices, and provides a polynomial time algorithm to compute the square roots of a given matrix. For details, see [3] (chapter VIII, p. 231).

The second one is based on the Cayley-Hamilton theorem. Let us denote by

$$\chi_M(\lambda) = \lambda^n + \alpha_{n-1}(M)\lambda^{n-1} + ... + \alpha_1(M)\lambda + \alpha_0(M)$$

the characteristic polynomial of a matrix $M \in \mathcal{M}_n(K)$. Since K is a field of characteristic 2, it is easy to prove that $\alpha_i(M^2) = \big(\alpha_i(M)\big)^2$ $(0 \leq i \leq n-1)$. Suppose now that A satisfies $A^2 = B$ for a given B. Then, from the Cayley-Hamilton theorem ($\chi_A(A) = 0$), we obtain the following formula:

$$A = \Big(\sqrt{\alpha_0(B)} \cdot I + \sqrt{\alpha_2(B)} \cdot B + \ldots\Big)\Big(\sqrt{\alpha_1(B)} \cdot I + \sqrt{\alpha_3(B)} \cdot B + \ldots\Big)^{-1}.$$

This method can only be used when $\alpha_1(B) \cdot I + \alpha_3(B) \cdot B + \ldots$ is invertible.

Note: The scheme can also be used in signature. To sign a message M, the basic idea is to compute x from $y = h(R||M)$ (as if we were deciphering a message), where h is a hash function and R is a small pad. If we succeed, (x, R) is the signature of M. If not (because the function is not a bijection), we try another pad R (for variants and details, see [9], where a similar idea is used).

12 Cryptanalysis of $[C_n]$

This section is given in appendix 3.

13 A Suggestion: the *HM* Scheme

The cryptanalysis of $[C_n]$ described in section 12 uses the fact that A and B commute when $B = A^2$. In order to avoid that very special algebraic property, we suggest to replace the transformation $B = A^2$ by the equation $B = A^2 + MA$, where M is a *secret* matrix randomly chosen in $\mathcal{M}_n(K)$.

The description of the obtained scheme – called *HM* – is exactly the same as for $[C_n]$. As in section 11, the transformation is generally not one-to-one, but the scheme can be used in a practical way because – as in the case of $[C_n]$ –, the number of pre-images of a given average matrix B remains under a reasonable limit. Table 4 below illustrates this fact (for a randomly chosen matrix M).

To obtain a practical scheme, one has to be able to solve the equation $A^2 + MA = B$ for a given matrix $B \in \mathcal{M}_n(K)$. There indeed exist a polynomial time algorithm to perform this computation (see [3], chapter VIII). The basic idea of this algorithm is to use the fact that $B = A^2 + MA$ implies $g(A) = 0$, where $g(\lambda) = \det(\lambda^2 + \lambda \cdot M - B)$ is a polynomial with *scalar* coefficients (notice that this property is a generalization of the Cayley-Hamilton theorem). The equation $g(A) = 0$ can be solved by using the Jordan reduction of matrices.

# pre-images	$n=2$	$n=3$	$n=4$	# pre-images	$n=2$	$n=3$	$n=4$
0	6	284	39552	16	0	0	72
1	8	112	12024	17	0	0	0
2	0	42	6576	18	0	0	12
3	0	32	2256	19	0	0	24
4	2	34	1868	20	0	0	24
5	0	0	960	21	0	0	0
6	0	0	972	22	0	0	24
7	0	0	168	23-25	0	0	0
8	0	2	324	26	0	0	36
9	0	0	48	27	0	0	0
10	0	2	144	28	0	0	6
11	0	0	96	29-33	0	0	0
12	0	4	162	34	0	0	4
13	0	0	56	35-39	0	0	0
14	0	0	72	40	0	0	8
15	0	0	48	> 40	0	0	0

Table 4: Number of pre-images for *HM* over $\mathbf{F}_2$ (toy examples)

	$n=2$	$n=3$	$n=4$
$p=2$	10 16 39	3 11 133	3 18 49
$p=3$	1 1 14	1 1 11	1 1 18
$p=31$	0 0 1	0 0 1	0 0 1
$p=127$	0 0 0	0 0 0	0 0 0

Table 5: Number of equations of
type 1 type 4
type 2
for *HM* over $K = \mathbf{F}_p$

The *HM* scheme seems less vulnerable to attacks based on affine multiple (*i.e.* on equations such that those of type (1), (2) or (4)), as shown in table 5 above (equations (4) are defined in appendix 3). However, we have made computations

(see table 6 below, in which the y_i variables have been replaced by explicit values) showing that equations of type (3) (defined in section 4) still exist, and also equations of "type (5)", defined by:

$$\sum \mu_{ij} x_i y_j + \sum \nu_{ij} x_i x_j + \sum \alpha_i x_i + \sum \beta_i y_i + \delta_0 = 0 \qquad (5)$$

HM	$n = 2$	$n = 3$	$n = 4$
Equations (5)	7	17	31
Equations (3)	9	30	58

Table 6: Number of linearly independent equations

Note: It may be noticed that $B = A^2$ implies the two following identities:

$$\begin{cases} AB - BA = AMA - MA^2 & \text{(type (5))} \\ A^2B - BA^2 = BMA - MAB & \text{(type (3))} \end{cases}$$

This explains – in part – the existence of such equations of types (5) and (3).

The existence of such equations threatens the *HM* scheme. In fact they make the cryptanalyst able to distinguish between a random quadratic transformation of K^{n^2} and a quadratic transformation corresponding to the *HM* scheme. This explains that we do not recommend the *HM* scheme. However, at the present, the existence of equations of type (5) and type (3) does not seem sufficient to break the scheme. Therefore, the question of the security of *HM* remains open...

14 Conclusion

Among cryptologists that have studied the problem, two main opinions arise as concerns public key schemes built with multivariate polynomials. Some of them think that most of these schemes should be vulnerable to attacks based on general principles, still to be found. According to others, the status of those many schemes can be compared to the one of most secret key algorithms: no relative proof of security is known, but the great flexibility for the choice among the possible variants of the schemes, together with the relative easiness for building efficient schemes that avoid known attacks, may support a certain confidence in the security of the schemes, at least – *a priori* – for those which do not seem too close to known cryptanalytic techniques.

The present article does not settle the question once and for all. Nevertheless, it gives arguments for both opinions. On the one hand, we have shown how to break some schemes for which no cryptanalysis had been given before. On the other hand, we have studied some simple and general ideas (removing equations, adding ones, introducing new variables...) that might – *a priori* – sensibly enforce the security of some asymmetric schemes. Interesting mathematical questions naturally arise: better understanding and detecting orthogonal polynomials, using a non commutative ring of matrices to generate multivariate equations on a (commutative) field, etc. If we had to take a strong line as concerns the unbroken schemes, our current opinion is that the most provocative schemes ($C^*\_\_$,

C^*_{-+}, *HM*) may be too close to known cryptanalysis to be recommended, but more complex schemes (such as HFE_{-+}) may be really secure... However, it is still too soon to have a definitive opinion, and we think that – above all – the important point is to go further into the understanding of the mysterious links between mathematics and the concepts of asymmetric cryptography and cryptanalysis.

References

1. D. Coppersmith, S. Winograd, *Matrix Multiplication via Arithmetic Progressions*, J. Symbolic Computation, 1990, vol. 9, pp. 251-280.
2. J.C. Faugere, *Rough evaluation* (personal communication).
3. F.R. Gantmacher, *The Theory of Matrices*, volume 1, Chelsae Publishing Company, New-York.
4. H. Imai, T. Matsumoto, *Algebraic Methods for Constructing Asymmetric Cryptosystems*, Algebraic Algorithms and Error Correcting Codes (AAECC-3), Grenoble, 1985, Lectures Notes in Computer Science n$^{\text{o}}$ 229, pp.108-119.
5. N. Koblitz, *Algebraic Aspects of Cryptography*, Algorithms and Computation in Mathematics, Volume 3, Springer, 1998.
6. R. Lidl, H. Niederreiter, *Finite Fields*, Encyclopedia of Mathematics and its applications, Volume 20, Cambridge University Press.
7. T. Matsumoto, H. Imai, *Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption*, Advances in Cryptology, Proceedings of EUROCRYPT'88, Springer-Verlag, pp. 419-453.
8. J. Patarin, *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88*, Advances in Cryptology, Proceedings of CRYPTO'95, Springer, pp. 248-261.
9. J. Patarin, *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP) : Two New Families of Asymmetric Algorithms*, Advances in Cryptology, Proceedings of EUROCRYPT'96, Springer, pp. 33-48.
10. J. Patarin, *Asymmetric Cryptography with a Hidden Monomial*, Advances in Cryptology, Proceedings of CRYPTO'96, Springer, pp. 45-60.
11. J. Patarin, L. Goubin, *Trapdoor One-way Permutations and Multivariate Polynomials*, Proceedings of ICICS'97, Springer, LNCS n$^{\text{o}}$1334, pp. 356-368.
12. J. Patarin, L. Goubin, *Asymmetric Cryptography with S-Boxes*, Proceedings of ICICS'97, Springer, LNCS n$^{\text{o}}$1334, pp. 369-380.

Appendix 1: First Cryptanalysis of C^*_-

Principle of the attack: Let P be the complete public form of C^*. We suppose that the first r public equations have been removed. Let $P_{(r+1)\ldots n}$ be the remaining part of P. The aim of the attack is to recover the public equations $P_{1\ldots r}$ and then to use the classical attack of [8]. Obviously, those equations can be found only modulo the vector space generated by all the public equations. The basic idea is to use the so-called *polar* form of P, defined by

$$Q(x,t) := P(x+t) - P(x) - P(t).$$

Description of the algorithm:
1) Choose randomly $t \neq 0$ and $x^{(0)}$.
2) Compute $z_{(r+1)...n} := Q_{(r+1)...n}(x^{(0)}, t)$.
3) Solve the equation

$$Q_{(r+1)...n}(x, t) = z_{(r+1)...n}, \qquad (**)$$

where x is the unknown. There are at least two solutions ($x^{(0)}$ and $x^{(0)} + t$) and at most 2.2^r solutions, because – for a given value $z_{1...r}$ – (among 2^r possible), the equation $Q(x, t) = z$ has 0 or 2 solutions (see the extended version for a proof).

Steps 1, 2 and 3 are repeated until we obtain the maximum number of solutions: 2.2^r (we use each time a different choice for $t \neq 0$ and $x^{(0)}$). The average number of necessary tries is estimated to be about 2^r.
4) Suppose we have found $t \neq 0$ and $x^{(0)}$ such that the $(**)$ equation has exactly 2.2^r solutions: $x^{(0)}, x^{(0)}+t, x^{(1)}, x^{(1)}+t, ..., x^{(2^r-1)}, x^{(2^r-1)}+t$. Let k be an integer such that $1 \leq k \leq r$. For half of the solutions, we have $Q_k(x, t) = 0$, and for the other half, we have $Q_k(x, t) = 1$, and this remains true if we consider only the subset $\left\{x^{(0)}, ..., x^{(2^r-1)}\right\}$ of the set of solutions. Therefore

$$\sum_{\nu=0}^{2^r-1} Q_k(x^{(\nu)}, t) = 2^{r-1},$$

which gives an equation of degree one on the $\frac{n(n-1)}{2} + 1$ coefficients of Q_k (this equation is the same for all the values k, $1 \leq k \leq r$).
5) By repeating steps 1-4 $\mathcal{O}(n^2)$ times, with different choices of $(x^{(0)}, t)$, we expect to find $\frac{n(n-1)}{2} + 1 - n$ equations on the coefficients of the Q_k ($1 \leq k \leq r$). This gives Q_1, ..., Q_r modulo the vector space generated by all the public equations.
6) Once Q is completely known, we deduce P_1, ..., P_n (there is a technical problem when the characteristic of K is 2, see the extended version), and the classical attack of [8] can be applied, so that C^*_- is broken for small r values. The complexity of this cryptanalysis is $\mathcal{O}(q^r)$, plus the complexity of the cryptanalysis of the original C^* scheme.

Note: This cryptanalysis uses deeply the fact that C^* is a permutation polynomial. A general theory about permutation polynomials, and the related notion of orthogonal systems of equations, can be found in [6], chapter 7.

Appendix 2: First Cryptanalysis of $[C]$

The security of the cryptosystem is based on the difficulty of solving the system $(\mathcal{S})$ (defined in section 9) of 4 quadratic equations in 4 variables over $K = \mathbf{F}_{2^m}$. Unfortunately, such a system can always be easily solved by using an algorithm based on Gröbner bases. At the present, the best implementations of Gröbner bases can solve any set of n quadratic equations with n variables

over any reasonable field K, when $n \leq 16$ approximately (cf [2]). Therefore, the original $[C]$ is not secure.

This first cryptanalysis shows that the parameter n must not be too small if we want to avoid attacks based on algebraic methods for solving systems of multivariate polynomial equations. That is why we are going to describe a generalization of the scheme to higher dimensions (for which Gröbner bases algorithms are inefficient) in section 11.

Appendix 3: Cryptanalysis of $[C_n]$

In this section, we describe a polynomial attack against the $[C_n]$ algorithm, which proves that this scheme is insecure. The key idea is to use the fact that $B = A^2$ implies $AB = BA$ (whereas two random matrices A and B do not commute in general). We begin by computing all the equations of type (1). The relation $AB = BA$ gives *a priori* n equations of this type. In fact, when we give explicit values to the y_i variables, we cannot obtain n independent linear equations on the x_i variables, since $AB = BA$ is also true when $B = P(A)$, where P is *any* polynomial in $K[X]$. The exact number of independent linear equations coming from $AB = BA$ is given by the following result of [3]:

Theorem 141 *The number N of linearly independent matrices that commute with the matrix B is given by the formula $N = n_1 + 3n_2 + ... + (2t-1)n_t$, where n_1, n_2, ..., n_t are the degrees of the non constant* invariant polynomials *of B.*

See [3], chapter VI, for the definition of the invariant polynomials, and chapter VIII for a proof of the theorem. In particular, we have $n \leq N \leq n^2$, with $N \simeq n$ in most of the cases. It remains – *a priori* – to perform an exhaustive search on $\simeq n$ variables to end the attack. In fact, we have made some simulations (see table 3 below) that suggest that there also exist many equations of type (2) (defined in section 4), and type (4) defined by:

$$\sum \gamma_{ij} x_i y_j + \sum \mu_{ij} y_i y_j + \sum \alpha_i x_i + \sum \beta_i y_i + \delta_0 = 0 \qquad (4)$$

	$n=2$	$n=3$	$n=4$
$p=2$	$^{10\ \ 16}_{39}$	$^{10\ \ 18}_{153}$	$^{17\ \ 32}_{292}$
$p=3$	$^{4\ \ 4}_{28}$	$^{9\ \ 9}_{89}$	$^{16\ \ 16}_{271}$
$p=31$	$^{3\ \ 3}_{14}$	$^{8\ \ 8}_{79}$	$^{15\ \ 15}_{254}$
$p=127$	$^{3\ \ 3}_{14}$	$^{8\ \ 8}_{79}$	$^{15\ \ 15}_{254}$

Table 3: Number of equations of $^{\text{type 1}\ \ \text{type 4}}_{\text{type 2}}$ for $[C_n]$ over $\mathbf{F}_p$

Note: For $p = 2$, on these examples, we obtain $(n + 1)$ (formally) linearly independent equations of type (1). This can be explained by the fact that – on the field $K = \mathbf{F}_2$ – the equations $B = A^2$ implies $\mathrm{tr}(B) = \mathrm{tr}(A)$.

These equations of type (1), (2) and (4) can be found by Gaussian reductions on a polynomial number of cleartext/ciphertext pairs. Therefore, the $[C_n]$ scheme is unlikely to be secure: by using all the found equations of type (1), (2) and (3), a cleartext is easily found by Gaussian reductions.