Trapdoor One-Way Permutations and Multivariate Polynomials

– Extended Version –

Jacques Patarin, Louis Goubin
Bull PTS, 68 route de Versailles - BP 45
78431 Louveciennes Cedex - France
email: J.Patarin@frlv.bull.fr
L.Goubin@frlv.bull.fr

Abstract

This article is divided into two parts. The first part describes the known candidates of trapdoor one-way permutations. The second part presents a new candidate trapdoor one-way permutation.

This candidate is based on properties of multivariate polynomials on finite fields, and has similar characteristics to T. Matsumoto, H. Imai, and J. Patarin's schemes.

What makes trapdoor one-way permutations particularly interesting is the fact that they immediately provide ciphering, signature, and authentication asymmetric schemes.

Our candidate performs excellently in secret key, and secret key computations can be implemented in low-cost smart-cards, *i.e.* without co-processors.

Key words : Trapdoor one-way permutations, multivariate polynomials, research of new asymmetric bijective schemes.

Notes:

- This paper is the extended version of the paper with the same title published at ICICS'97.
- In this extended version, we have taken into account the recent results of [5].

Part I
Known candidates

1 Introduction

Nobody can deny that the idea of trapdoor one-way permutation plays a very important role in cryptography. Many theoretic schemes use this concept as an "elementary block". Moreover, any candidate trapdoor one-way permutation can easily be transformed into a scheme of asymmetric cryptography, for ciphering, signature, as well as authentication.

Amazingly enough, no widespread paper exists that describes all the known candidates at present. As a result, many people think for example that RSA is the only explicit and available candidate today. As we will quickly see in the first part of this paper, it is only *almost* true. In fact, one can obtain many variants of RSA: by taking even exponents and a modified message space to keep bijectivity (Rabin-Williams), by using polynomial permutations that are different from the modular exponentiations (Dickson polynomials for instance), or by performing the computations in other groups (such as elliptic curves). There are also much less well known candidates, very different from RSA, that are based on public forms given by multivariate equations (the original idea was first presented by T. Matsumoto and H. Imai).

integers. This link between the factorisation of the integers and the concept of trapdoor one-way permutation may seem surprising. This is a strong motivation to look for other ways of designing candidate trapdoor one-way permutations.

In this paper, we present a new example of such a candidate, that we have called D^{**}. One of the main interests of D^{**} lies in the fact that secret key computations are easy to implement: they are about 100 times faster than in 512 bits-RSA, and they require about 5 times less RAM. Therefore, D^{**} can be implemented in a smartcard without arithmetic co-processor (on the contrary, public key computations are supposed to be performed on a personal computer).

The security of D^{**}, as well as the security of other algorithms of the same family, cannot be related to a difficult problem as easily as RSA-like cryptosystems. Nevertheless, one can hope that these algorithms show interesting ways to build new candidates, or to discover new ideas in asymmetric cryptography.

2 Trapdoor one-way permutations

Let us recall the definition of a one-way function:

Definition: Let $f : A \to B$ be a function. f is said to be *one-way* if:

(i) Given $x \in A$, it is computationally easy to compute $y = f(x)$.

(ii) Given $y \in_f B$, it is computationally hard to compute $x \in A$ such that $f(x) = y$.

Note: In this definition, $\in_f$ means that y is "randomly" chosen in $f(A)$, where "randomly" means here that the probability of obtaining a value y is exactly:

$$\frac{|\{x \in A,\ f(x) = y\}|}{|A|}$$

Although many functions are thought to verify these two properties (they are called "candidate one-way functions"), nobody has ever *proven* that one-way functions exist. Moreover, in this paper, we will focus on a special class of them, namely the *trapdoor* one-way functions, which we define as follows:

Definition : Let $f : A \to B$ be a function. f is said to be trapdoor one-way if:

(i) f is a one-way function.

(ii) There is a secret information s such that, given $y \in_f B$ and s, it is computationally easy to compute $x \in A$ such that $f(x) = y$.

More precisely, we will only consider trapdoor one-way *permutations*, *i.e.* trapdoor one-way functions which are also bijective.
As we mentioned above, no function has ever been proven to be a trapdoor one-way permutation. If many "candidate one-way functions" are known, on the opposite, few candidate "trapdoor one-way functions" are known (they give essentially all the known asymmetric cryptosystems), and very few candidate "trapdoor one-way permutations" are known. We will now describe quickly all the candidate trapdoor one-way permutations we are aware of.

2.1 RSA

This is the most famous trapdoor one-way permutation. It was designed by Rivest, Shamir and Adleman in 1978 (cf [24]).
Suppose that n is the product of two large primes p and q, and let e be an integer. We consider the following function:

$$f : \begin{cases} \mathbf{Z}/n\mathbf{Z} \to \mathbf{Z}/n\mathbf{Z} \\ x \mapsto x^e \end{cases}$$

f is expected to be a trapdoor one-way permutation, whose corresponding secret information is the factorisation $n = pq$.
With this secret information, it is very easy to invert f:

$$f^{-1}(y) \equiv y^d \bmod n,$$

where $ed \equiv 1 \bmod \lambda(n)$ (d can be easily computed with Euclide's algorithm).

Note : It is obvious that:

(i) f is a permutation.

(ii) f is a trapdoor function.

What remains unclear is whether f is one-way. It has not been proven that factoring n is computationally hard (or that finding the secret exponent d, which can be proven equivalent to factoring n, is computationally hard). Furthermore, even if it is true, it is not clear whether we need factoring n (or computing d) to be able to compute $f^{-1}(y)$. These are two famous open problems.

2.2 Rabin-Williams

As we saw in the previous section, breaking RSA with modulus n has not be proven to be as difficult as factoring n. In 1979, Rabin (cf [23]) introduced the following modification of the scheme: instead of choosing e coprime to $\lambda(n)$, one can take $e = 2$. It can be shown that computing square roots is as difficult as factoring n. Unfortunately, the obtained function is no longer a permutation, which may make the decrypted messages ambiguous. One classical way to solve this problem is adding redundancy in the cleartext.
However, in 1980, Williams (cf [27]) showed a more elegant way to eliminate this problem: p and q are chosen so that $p \equiv 3 \bmod 8$ and $q \equiv 7 \bmod 8$. We take $n = pq$, and a small integer s such that $(\frac{s}{n}) = -1$ (where $(\frac{\cdot}{\cdot})$ is the Jacobi symbol). n and s are public. Let $d = \frac{1}{2}(\frac{1}{4}(p-1)(q-1)+1)$. We define:

$$f : \begin{cases} \mathbf{Z}/n\mathbf{Z} \to Q_n \times \{0,1\} \times \mathbf{Z}/2\mathbf{Z} \\ x \mapsto \begin{cases} (x^2, 0, x \bmod 2) & \text{if } (\frac{x}{n}) = 1 \\ ((sx)^2, 1, sx \bmod 2) & \text{if } (\frac{x}{n}) = -1 \end{cases} \end{cases}$$

where Q_n is the set of quadratic residues in $\mathbf{Z}/n\mathbf{Z}$.
This function is a trapdoor permutation and it can be proven that finding a cleartext from a random ciphertext is as difficult as factoring.

Note: In 1985, Williams (cf [28]) extended this idea to $e = 3$ and $\mathbf{Z}[\omega]$ for the message space instead of $\mathbf{Z}$ (where ω is a primitive cube root of unity). In this public-key scheme, computing cleartexts from random ciphertexts is also provably as intractable as factoring n. In 1992, Loxton, Khoo, Bird and Seberry ([10]) gave another variant, with another choice for the complete set of residues used in defining the message space.

2.3 The Kurosawa-Itoh-Takeuchi cryptosystem

In [9], Kurosawa, Itoh and Takeuchi proposed the following trapdoor one-way permutation. Let $n = pq$ stands for the product of two large primes p and q, and let c be an integer such that $(\frac{c}{p}) = (\frac{c}{q}) = -1$ (where $(\frac{\cdot}{\cdot})$ is the Legendre symbol). If $D_{n,c} = \{y \in \mathbf{Z}/n\mathbf{Z},\ y^2 - 4c \in Q_n^*\}$ (where Q_n^* is the set of non-zero quadratic residues in $\mathbf{Z}/n\mathbf{Z}$), we define:

$$f : \begin{cases} (\mathbf{Z}/n\mathbf{Z})^* \to D_{n,c} \times \{0,1\} \times \{0,1\} \\ x \mapsto (x + (c/x) \bmod n,\ s,\ t) \end{cases}$$

where

$$s = \begin{cases} 0 & \text{if } (\frac{x}{n}) = 1 \\ 1 & \text{if } (\frac{x}{n}) = -1 \end{cases} \quad \text{and} \quad t = \begin{cases} 0 & \text{if } (c/x \bmod n) > x \\ 1 & \text{if } (c/x \bmod n) < x. \end{cases}$$

This function is a trapdoor permutation and it can be proven that finding a cleartext from a random ciphertext is as difficult as factoring.

In [18], Pascal Paillier developped a new one-way trapdoor permutation over $\mathbf{Z}_{n^2}^*$. More precisely, let $n = pq$ stands for the product of two large primes p and q, let $\lambda = \lambda(n) = \text{lcm}(p-1, q-1)$ and let L be the function defined over $\mathcal{S}_n = \{u < n^2,\ u = 1 \bmod n\}$ by:

$$\forall u \in \mathcal{S}_n,\ L(u) = \frac{u-1}{n}.$$

If g is chosen such that $\gcd(\text{L}(g^\lambda \bmod n^2),\ n) = 1$, we define:

$$f : \begin{cases} \mathbf{Z}_{n^2}^* \to \mathbf{Z}_{n^2}^* \\ x = x_1 + nx_2 \mapsto g^{x_1} {x_2}^n \bmod n^2 \end{cases}$$

The function f is a trapdoor permutation, and it can be proven that it is one-way if and only if the so-called RSA$[n, n]$ problem (*i.e.* extracting n-th roots modulo n, where $n = pq$ is of unknown factorisation) is hard.

2.5 Dickson polynomials

In [16] and [17] (see also [11]), Winfried Müller and Rupert Nöbauer developed a variant of RSA that makes use of Dickson polynomials, instead of the x^e monomial. This idea was generalized by Rudolph Lidl (see [12]) and W. Müller (see [15]). Basically, the schemes use the Dickson polynomials g_k defined by:

$$g_k(X) = \sum_{i=0}^{\lfloor \frac{k}{2} \rfloor} \frac{k}{k-i} \binom{k-i}{i} (-1)^i x^{k-2i}.$$

Let $n = \prod_{i=1}^{r} p_i^{\alpha_i}$ and $v(n) = \text{lcm}(p_i^{\alpha_i - 1}(p_i^2 - 1),\ 1 \le i \le r)$. It can be proven that g_k is a permutation of $\mathbf{Z}/n\mathbf{Z}$ if and only if $\gcd(k, v(n)) = 1$, and that – in that case – the inverse of g_k is g_t, where $kt \equiv 1 \bmod v(n)$. From this property, it is easy to derive an analogue of RSA. Moreover, the only known method to invert g_k needs the factorisation of n, so that we have another candidate trapdoor one-way permutation based on the factoring problem.

Notes :

1. The Dickson polynomials are also known as Chebyshev polynomials of the first kind.
2. In 1993, the scheme of Müller and Nöbauer was re-invented (with minor differences) by P.J. Smith, who called it LUC (see [25] and [26]). This cryptosystem is formulated in terms of Lucas sequences. Some variations of LUC were also developed as (non bijective) analogies to the ElGamal scheme. Daniel Bleichenbacher, Wieb Bosma and Arjen K. Lenstra (see [1]) showed that – because of the deep links between Lucas sequences and exponentiation – all these variations of LUC, as well as RSA, are vulnerable to subexponential time attacks.

2.6 The Gong-Harn cryptosystem

In [7], Gong and Harn proposed the following trapdoor one-way permutation. Let $n = pq$ stands for the product of two large primes p and q, and let e be an integer such that

$$\gcd(e, p^2 - 1) = \gcd(e, p^3 - 1) = \gcd(e, q^2 - 1) = \gcd(e, q^3 - 1) = 1.$$

If a and b are two elements of $\mathbf{Z}/n\mathbf{Z}$, we define $s_e(a, b)$ and $s_{-e}(a, b)$ by the following equation:

$$X^3 - aX^2 + bX - 1 = (X - \alpha_1)(X - \alpha_2)(X - \alpha_3)$$

$$\Rightarrow (X - \alpha_1^e)(X - \alpha_2^e)(X - \alpha_3^e) = X^3 - s_e(a, b)X^2 + s_{-e}(a, b)X - 1.$$

We then consider:

$$f : \begin{cases} (\mathbf{Z}/n\mathbf{Z})^2 \to (\mathbf{Z}/n\mathbf{Z})^2 \\ (x_1, x_2) \mapsto (s_e(x_1, x_2), s_{-e}(x_1, x_2)). \end{cases}$$

This function is a trapdoor permutation and it can be proven that finding a cleartext from a random ciphertext is as difficult as factoring.

Another way to obtain analogues of RSA is to use elliptic curves over the ring $\mathbf{Z}/n\mathbf{Z}$ instead of the ring $\mathbf{Z}/n\mathbf{Z}$ itself to perform the computations. For any integer n, we denote by $E_n(a,b)$ the following elliptic curve:

$$E_n(a,b) = \{(x,y) \in (\mathbf{Z}/n\mathbf{Z})^2,\ y^2 \equiv x^3 + ax + b \bmod n\}.$$

- In 1991, Kenji Koyama, Ueli M. Maurer, Tatsuaki Okamoto and Scott A. Vanstone (see [8]) proposed the following scheme: they choose two prime numbers p and q such that $p \equiv q \equiv 2 \bmod 3$, and an integer e coprime to $(p+1)(q+1)$. As in RSA, e, n are public, and p, q are secret. Each message is represented by an element $(x,y) \in (\mathbf{Z}/n\mathbf{Z})^2$. The encryption function is defined by:

$$f : \begin{cases} (\mathbf{Z}/n\mathbf{Z})^2 \to (\mathbf{Z}/n\mathbf{Z})^2 \\ (x,y) \mapsto e.(x,y) \end{cases} \qquad \text{the calculus being performed in } E_n(0, y^2 - x^3 - ax).$$

  With the secrets d, it is very easy to invert f:

$$f^{-1}(x',y') = d.(x',y') \qquad \text{the calculus being performed in } E_n(0, y'^2 - x'^3 - ax'),$$

  where $ed \equiv 1 \bmod \operatorname{lcm}(p+1, q+1)$.

  Notes :

  1. A variant chooses $p \equiv q \equiv 3 \bmod 4$ and performs the encryptions and decryptions in $E_n(a,0)$ instead of $E_n(0,b)$.
  2. In the same way, we obtain an elliptic curve based analogue of the Rabin scheme.

  As a result, this gives new candidate trapdoor one-way permutations, whose security is again based on the difficulty of factoring n. Moreover, these schemes seem to be more secure than RSA (or Rabin) against attacks without factoring, such as low multiplier attacks.

- In 1993, N. Demytko (see [3]) proposed another elliptic curve cryptosystem, whose security is also based on the difficulty of factoring $n = pq$, where p and q are secret prime integers. In this scheme, a *fixed* elliptic curve $E_n(a,b)$, with $\gcd(4a^3+27b^2, n) = 1$, and an integer e are chosen and made public. Each message is represented by an element $x \in \mathbf{Z}/n\mathbf{Z}$. The ciphertext $x' \in \mathbf{Z}/n\mathbf{Z}$ is defined as the first coordinate of the point $e.P \in E_n(a,b)$, where P is a point of the elliptic curve $E_n(a,b)$ whose first coordinate is x. There are explicit formulas giving x' in terms of x (and requiring neither the second coordinate of P, nor the secret parameters p and q), so that the encryption function above is well defined and can be performed by anyone. Moreover, when e is suitably chosen, an integer d can be computed with the secret parameters p and q, so that the cleartext of $x' \in \mathbf{Z}/n\mathbf{Z}$ is the first coordinate of $d.Q \in E_n(a,b)$, where Q is a point of $E_n(a,b)$ whose first coordinate is x'.

  The security of these candidate trapdoor one-way permutations also relies on the difficulty of factoring large composite numbers.

2.8 The C^* scheme

In 1988, Hideki Imai and Tsutomu Matsumoto proposed a very different public key scheme, called C^* (see [14]), that is based on multivariate polynomials over a finite field. The basic idea is to represent a message by an element $x \in K^n$, where $K = GF(2^m)$ is a public finite field, and n is a public integer. An integer θ such that $\gcd(1 + 2^{m\theta}, 2^{mn} - 1) = 1$, and an extension $\mathcal{L}_n$ of degree n over K are also public. The encryption function is defined by:

$$f : \begin{cases} K^n \to K^n \\ x \mapsto t(s(x)^{1+2^{m\theta}}) \end{cases}$$

where $s : K^n \to \mathcal{L}_n$ and $t : \mathcal{L}_n \to K^n$ are secret affine permutations. As f can be given by n public polynomials in n indeterminates over K, anyone can encrypt a message. It is easy to see that f is a trapdoor permutation. However, the C^* scheme was broken in 1995 by Jacques Patarin (see [19]), and therefore is not one-way.

no longer bijective.

2.9 ABC

The C^* scheme we described in the previous section is not the only attempt of Matsumoto and Imai to design trapdoor one-way permutations (also called ABC, for Asymmetric Bijective Cryptosystems). In 1985 (see [13]), they proposed three schemes – called A, B and C – based on multivariate polynomials over finite fields.

- The first one is the same as the C^* scheme described in the section above.
- In the B scheme, $K = GF(2)$ and an integer n are public. The encryption function is:

$$f : \begin{cases} K^n \to K^n \\ x \mapsto \begin{cases} t((s(x) + c - 1) \bmod (2^n - 1) + 1) & \text{if } x \neq 0 \\ 0 & \text{if } x = 0 \end{cases} \end{cases}$$

  where $s : K^n \to E$ and $t : E \to K^n$ are secret linear bijections, $E = \{k,\ 0 \leq k < 2^n\} = \left\{ \sum_{i=0}^{n-1} \alpha_i 2^i \right\}$ is considered as a vector space of dimension n over K, and c is a positive integer whose binary expression has small Hamming weight. The public-key is an "and-exclusive or" array pattern for the n-uple of n-variate sparse polynomials over K that represent f. We do not know if any cryptanalytic work has been done against this B scheme. Its security is – as far as we know – an open problem.
- In the C scheme, $K = GF(2^m)$ is public, and the encryption function is:

$$f : \begin{cases} K^4 \to K^4 \\ x \mapsto t(s(x)^2) \end{cases}$$

  where $s : K^4 \to GL_2(K)$ and $t : GL_2(K) \to K^4$ are secret linear bijections, the set $GL_2(K)$ of 2×2 matrices over K being considered as a vector space of dimension 4 over K. The public key is the 4-uple of 4-variate quadratic polynomials over K that represent f. Moreover, with some minor changes, the scheme can be made bijective. However, in [22], we have presented a way to break this C scheme (the idea is to use the fact that $AB = BA$ when $B = A^2$ and A and B are two matrices).

Part II
Presentation of D^*

In part III, we will describe a new candidate trapdoor one-way permutation. However, in order to describe this candidate, called D^{**}, we need first to describe a scheme, called D^*. As we will see, this scheme D^* is not secure, but will be useful in part III. D^* and D^{**} have many analogies with the C^* scheme of section 2.5: the aim is to avoid the attacks and – in the same time – to keep the bijective properties.

3 D^*: a new tool

This section is devoted to the description of the D^* scheme used in encryption mode.

3.1 Representation of the message

A finite field $K = GF(q)$ is *public*, where $q = p^m$, m is odd, and p is a prime number such that $p \equiv 3 \bmod 4$ and p is not too small (for example $p = 251$; this point will be explained in section 3.6). Each message M is represented by n elements of K, where n is an *odd* and *public* integer (for example $n = 9$).

will be useful.

Moreover, we choose the representation x of M in the following message space:

$$\mathcal{M} = \Big\{ x = (x_1, ..., x_n) \in K^n,\ \exists k,\ 1 \leq k \leq n,\ x_k \in K' \text{ and } (\forall i < k,\ x_i = 0) \Big\}$$

where K' is a complete set of residues of $K^*/\{\pm 1\}$.

Notes:

1. If $m = 1$, *i.e.* $K = GF(p)$, we can choose for example $K' = \{x.1_K,\ 1 \leq x \leq \frac{p-1}{2}\}$.
2. More generally, if $(e_1, ..., e_m)$ is an arbitrary base of K over $GF(p)$, we can choose:

$$K' = \Big\{ x = \sum_{i=1}^{m} x_i e_i \in K,\ \exists k,\ 1 \leq k \leq m,\ \Big(1 \leq x_k \leq \frac{p-1}{2}\Big) \text{ and } (\forall i < k,\ x_i = 0) \Big\}.$$

3. It is easy to verify on the examples above that:

$$|K'| = \frac{p-1}{2}(p^{m-1} + p^{m-2} + ... + p + 1) = \frac{p^m - 1}{2} = \frac{q-1}{2}$$

$$|\mathcal{M}| = \frac{q-1}{2}(q^{n-1} + q^{n-2} + ... + q + 1) = \frac{q^n - 1}{2}.$$

4. Any complete set of residues of $(K^n \setminus \{0\})/\{\pm 1\}$ can be chosen as the message space $\mathcal{M}$.

3.2 Encryption of $x \in \mathcal{M}$

The scheme also uses:

1. An extension $\mathcal{L}_n$ of degree n over K.
2. Two linear *secret* bijections $s : K^n \to \mathcal{L}_n$ and $t : \mathcal{L}_n \to K^n$. In a basis, these two linear permutations can be written as n polynomials in n variables over K, and of total degree one.

Note: $\mathcal{L}_n$ can be made public without reducing the security of the scheme, because changing $\mathcal{L}_n$ is equivalent to making other choices for s and t, so that $\mathcal{L}_n$ can be considered as a fixed extension.

With the preceeding notations, the ciphering algorithm can be described as follows. $y = F(x)$ is defined as the only element of $\{ + t(s(x)^2), -t(s(x)^2)\}$ that belongs to $\mathcal{M}$ (this element exists and is unique, by construction of $\mathcal{M}$). Since s and t are of total degree one over K, $t(s(x)^2)$ can be given in a basis by n quadratic polynomials P_1, ... ,P_n in n variables, whose coefficients belong to K.
These polynomials are made public, so that anyone can easily encrypt a message, by using the following equations to compute $y = (y_1, ..., y_n) = F(x)$ from $x = (x_1, ..., x_n) \in \mathcal{M}$:

$$\begin{cases} y = (y_1, ..., y_n) \in \mathcal{M} \\ y_1 = \pm P_1(x_1, ..., x_n) \\ ... \\ y_n = \pm P_n(x_1, ..., x_n) \end{cases}$$

3.3 Decryption of $y \in \mathcal{M}$

Under the hypothesis we have made ($p \equiv -1 \bmod 4$, m odd and n odd), it is easy to prove that, for any $y \in \mathcal{M}$ and $x \in K^n$:

1. If $t^{-1}(y)$ is a quadratic residue in $\mathcal{L}_n$, then

$$F(x) = y \Leftrightarrow t(s(x)^2) = y \Leftrightarrow x = \pm s^{-1}\Big((t^{-1}(y))^{\frac{q^n+1}{4}}\Big)$$

and we can choose the sign so as to ensure $x \in \mathcal{M}$.

$q^n \equiv -1 \bmod 4 \Rightarrow (-1)^{\frac{q^n-1}{2}} = -1 \Rightarrow -1$ is not a quadratic residue in $\mathcal{L}_n$). As a result:

$$F(x) = y \Leftrightarrow t(s(x)^2) = -y \Leftrightarrow x = \pm s^{-1}\Big((- t^{-1}(y))^{\frac{q^n+1}{4}}\Big) \Leftrightarrow x = \pm s^{-1}\Big((t^{-1}(y))^{\frac{q^n+1}{4}}\Big)$$

and the sign can also be chosen so that $x \in \mathcal{M}$.

Therefore, the encryption function F is a permutation from $\mathcal{M}$ to $\mathcal{M}$, whose inverse F^{-1} is easy to compute for anyone who knows the secret linear permutations s and t: for any $y \in \mathcal{M}$, $x = F^{-1}(y)$ is characterized by the following formula:

$$\begin{cases} x \in \mathcal{M} \\ x = \pm s^{-1}\Big((t^{-1}(y))^{\frac{q^n+1}{4}}\Big). \end{cases}$$

3.4 Complexity of the encryption and decryption algorithms

Encryption Obviously, using the public polynomials P_1, ..., P_n to encrypt a message requires $\leq \frac{n^3}{2}$ multiplications in $K = GF(q)$ and $\leq \frac{n^3}{2}$ additions in K. Since the complexity of a multiplication in K is $\mathcal{O}((\log q)^2)$, the encryption algorithm has a complexity $\mathcal{O}(n^3 m^2 (\log p)^2)$.

Note: Asymptotically, the complexity of a multiplication in $K = GF(q)$ is $\mathcal{O}(\log q. \log\log q)$ for very large q, but for our practical values of q, the algorithms are in $\mathcal{O}(\log^2 q)$, since the size of q is reasonable.

Decryption The decryption function is given by the following formula:

$$x = \pm s^{-1}\Big((t^{-1}(y))^{\frac{q^n+1}{4}}\Big).$$

Each linear transformation requires $\leq n^2$ multiplications and additions in $K = GF(q)$.
For the exponentiation, we can use the following identity:

$$\frac{q^n+1}{4} = \frac{q+1}{4}\Big[q(q-1)\sum_{i=0}^{\frac{n-3}{2}} q^{2i} + 1\Big].$$

If we use a normal base for $\mathcal{L}_n$ (*i.e.* a base that can be written $(\beta, \beta^q, \beta^{q^2}, ..., \beta^{q^{n-1}})$ for some $\beta \in \mathcal{L}_n$), we see that the complexity of the evaluation of the q^k-th power of an element of $\mathcal{L}_n$ can be neglected as compared to the complexity of the multiplication of two elements in $\mathcal{L}_n$.
Moreover, we can use the following remark: if we write the binary representation of $\frac{n-3}{2}$ as:

$$\frac{n-3}{2} = \sum_{\nu=1}^{N} 2^{\beta_\nu} \qquad (\beta_1 < \beta_2 < ... < \beta_N)$$

we can obtain the following identity:

$$\sum_{i=0}^{\frac{n-3}{2}} q^{2i} = \Big(\Big(...\Big(\Big(\prod_{j=\beta_{N-1}+1}^{\beta_N}(q^{2^j}+1) + q^{2.2^{\beta_N}}\Big)\prod_{j=\beta_{N-2}+1}^{\beta_{N-1}}(q^{2^j}+1) + q^{2(2^{\beta_N}+2^{\beta_{N-1}})}\Big)\prod_{j=\beta_{N-3}+1}^{\beta_{N-2}}(q^{2^j}+1)$$

$$+ ...\Big)\prod_{j=\beta_1+1}^{\beta_2}(q^{2^j}+1) + q^{2(2^{\beta_N}+...+2^{\beta_2})}\Big)\prod_{j=1}^{\beta_1}(q^{2^j}+1) + q^{2(2^{\beta_N}+...+2^{\beta_1})},$$

so that evaluating the $\sum_{i=0}^{\frac{n-3}{2}} q^{2i}$-th power of an element of $\mathcal{L}_n$ requires $\leq N + \beta_N \leq 3\log n$ multiplications in $\mathcal{L}_n$ and $\leq 3\log n$ evaluations of q^k-th powers in $\mathcal{L}_n$.

estimate the running time of their C^* scheme (cf [14], page 428).

In conclusion, the decryption algorithm requires at most $3n^2(\log n + \log q)$ multiplications or q^k-th exponentiations in $\mathcal{L}_n$, so that the complexity of the decryption algorithm is $\mathcal{O}(3(mn)^2(m \log p + \log n)(\log p)^2)$.

3.5 Complexity of solving quadratic systems in a field

To attack the cryptosystem described in this section, one of the most obvious ideas is trying to solve the following quadratic system:

$$\begin{cases} P_1(x_1, ..., x_n) = \pm y_1 \\ ... \\ P_n(x_1, ..., x_n) = \pm y_n \end{cases}$$

to find the cleartext x of a given ciphertext y.
However, we have proven that – whatever the field K may be – the *general* problem of solving a randomly selected system of multivariate quadratic equations over K is NP complete. This result was already known for $K = GF(2)$ (cf [6] page 251). Our proof of the general case is given in the appendix.

Notes:

1. In our scheme, it is not possible to choose $p = 2$, because $P_1, ..., P_n$ would be of total degree one, and thus F would be a simple linear transformation, and so could be very easily inverted by gaussian reductions.

2. The problem the cryptanalyst has to cope with is a *particular* instance of a quadratic system over K. Therefore, the argument above does not prove that breaking the system is a NP-complete problem. Moreover, a classical theoretical argument of G. Brassard shows that breaking an encryption scheme is never a NP-complete problem.

3.6 The affine multiple attack

Another attack, which is very general, was described in [20]. It can be used against schemes based on a univariate polynomial transformation hidden by secret affine bijective transformations.
This attack is based on the following fact : if f is a univariate polynomial over a finite extension L of a finite field K, then by using a general algorithm (see for example [2]), one can compute an "affine multiple" of the polynomial $f(a) - b$, *i.e.* a polynomial $A(a, b) \in L[X, Y]$ such that:

1. Each solution of $f(a) = b$ is also a solution of $A(a, b) = 0$.

2. $A(a, b)$ is an affine function of a when written in a basis over K.

In the case of the D^* scheme, $L = \mathcal{L}_n$ and $f(a) = a^2$. It can be proven that any non-zero affine multiple $A(a, b)$ of f is at least of degree $\frac{p-3}{2}$ with respect to b. We have taken it for granted that p is not too small (a typical example is $p = 251$). With this hypothesis, the affine multiple attack does not threat the D^* scheme, because there is no practical way to compute $A(a, b)$.

4 First cryptanalysis of D^*

In this section, we prove that D^* is not secure.

- The cryptanalysis is based on the following identity:

$$uv = \frac{(u + v)^2 - (u - v)^2}{2},$$

which is valid because the characteristic of the field K is not 2. As a result:

$$\frac{F(x + x') - F(x - x')}{2} = \pm t(s(x) \cdot s(x')).$$

Therefore, $\phi(x, x') = t(s(x) \cdot s(x'))$ is given by n *public* bilinear forms with coefficients in K.

that:

$$\forall x, x' \in K^n,\ C(\phi(x, x')) = \phi(D(x), x').$$

This vector space is at least of dimension n, because we can choose, for *any* $\lambda \in \mathcal{L}_n$:

$$\begin{cases} D(x) = s^{-1}(\lambda \cdot s(x)) \\ C(y) = t(\lambda \cdot t^{-1}(y)). \end{cases}$$

For simplicity, let us assume that the dimension is *exactly* n (we have made some simulations that confirm this property).

Since the set of solutions for C depends on n free variables, we can call these variables Λ_1, ..., Λ_n, and denote by C_Λ the solution with parameter $\Lambda = (\Lambda_1, ..., \Lambda_n)$.

- We then compute the vector space of all linear transformations E from K^n to K^n such that:

$$C_{E(\Lambda)}(\tilde{y}) = C_{E(\tilde{y})}(\Lambda).$$

Here again, we find a vector space of dimension at least n.

Note: This is due to the fact that, by definition, $C_\Lambda(\tilde{y}) = t(\theta(\Lambda) \cdot t^{-1}(\tilde{y}))$, where θ is an unknown linear transformation from K_n to $\mathcal{L}_n$. Therefore, for any $\mu \in \mathcal{L}_n$, we can choose $E = \theta^{-1}(\mu \cdot t^{-1})$, and so obtain a solution.

Let E_0 be such a solution, and let $*$ be the operation such that, by definition:

$$\Lambda * \tilde{y} = \tilde{y} * \Lambda = C_{E_0(\Lambda)}(\tilde{y}).$$

Notes:

1. t and μ are still unknown, but $*$ has been found out.
2. By construction, it is easy to see that:

$$\exists \mu \in \mathcal{L}_n,\ \mu \neq 0,\ \forall \Lambda \in K^n,\ \forall \tilde{y} \in K^n,\ \Lambda * \tilde{y} = t(\mu \cdot t^{-1}(\Lambda) \cdot t^{-1}(\tilde{y})).$$

- We now compute, with the square-and-multiply principle (applied to the $*$ law):

$$\tilde{y}^{*(\frac{q^n+1}{4})} = \underbrace{\tilde{y} * ... * \tilde{y}}_{\frac{q^n+1}{4} \text{ times}} = t\left(\mu^{\frac{q^n+1}{4}-1} \cdot t^{-1}(\tilde{y})^{\frac{q^n+1}{4}}\right) = \pm t(\mu^{\frac{q^n+1}{4}-1} \cdot s(x)).$$

As a result, a linear transformation W from K^n to K^n exists, such that any cleartext/ciphertext pair (x, y) satisfies the following equation:

$$y^{*(\frac{q^n+1}{4})} = \pm W(x).$$

Moreover, W can be easily found by gaussian reductions on a few cleartext/ciphertext pairs. More precisely, let $(x[1], y[1])$, ..., $(x[k], y[k])$ be k cleartext/ciphertext pairs. According to the previous remark, there exist k elements of $\{-1, +1\}$, denoted by ϵ_1, ..., ϵ_k, such that:

$$\forall j,\ 1 \leq j \leq k,\ \epsilon_j x[j] - W^{-1}(y[j]^{*(\frac{q^n+1}{4})}) = 0.$$

As a result, we have nk equations (n equations for each value of j) and $n^2 + k$ unknown values (the n^2 coefficients of W^{-1} and the k variables ϵ_j). Moreover, if we suppose that $k \geq \frac{n^2}{n-1}$, we have $nk \geq n^2 + k$, so that the system can be solved.

solutions (which are opposite from each other) if the conditions $\epsilon_j = \pm 1$ $(1 \leq j \leq k)$ are taken into account. Moreover, to have a unique solution, we can suppose $\epsilon_1 = 1$ for example.

- After $*$ and W have been found, decrypting any ciphertext is easy, since:

$$x = \pm W^{-1}(y^{*(\frac{q^n+1}{4})}).$$

5 E^* cryptosystem - Description and cryptanalysis

- We call E^* the algorithm similar to D^*, but with $b = a^3$ instead of $b = a^2$, so that the encryption function G is given by:

$$y = G(x) = t(s(x)^3).$$

- As for D^*, the public key consists of n polynomials in n variables over K. For E^*, these polynomials are *cubic*.

- If we choose $p \equiv 2 \bmod 3$, m odd and n odd, then $\gcd(3, q^n - 1) = 1$, so that $b = a^3 \Leftrightarrow a = b^h$, where h is the inverse of 3 modulo $q^n - 1$. Under these asumptions, G is a trapdoor permutation of K^n, and the decryption function is given by:

$$x = G^{-1}(y) = s^{-1}(t^{-1}(y)^h).$$

- However, we are going to prove that the permutation G is not one-way. Moreover, the cryptanalysis is similar to that of D^*, and is based on the following identity:

$$uvw = \frac{(u+v+w)^3 + (u-v-w)^3 - (u-v+w)^3 - (u+v-w)^3}{24},$$

which is valid as soon as the characteristic of K is greater than 3. Thus:

$$\frac{G(x+x'+x'') + G(x-x'-x'') - G(x-x'+x'') - G(x+x'-x'')}{24} = t(s(x) \cdot s(x') \cdot s(x'')).$$

Therefore, $\psi(x, x', x'') = t(s(x) \cdot s(x') \cdot s(x''))$ is given by n public cubic forms with coefficients in K.

A $*$ law can then be derived exactly as in section 4 – by writing $s(x) \cdot s(x') \cdot s(x'')$ as $s(x) \cdot [s(x') \cdot s(x'')]$ – and finally, with a few cleartext/ciphertext pairs, any message can be easily decrypted.

6 Cryptanalysis of more general polynomial transformations

The algorithm used for the first cryptanalysis of D^* (*i.e.* with the polynomial $f(X) = X^2$), or for E^* (*i.e.* with the polynomial $f(X) = X^3$), can also be extended to any polynomial transformation $f(X)$ as long as the degree D of f is less than the characterisctic p of the field K.

Note: Moreover, when the degree D of the hidden polynomial f is smaller than p, then this degree D should be very small because it will also be the total degree of the public equations and the size of the public key must be reasonable.

So, let us assume $D < p$, and let $f(X) = \sum_{i=1}^{D} \alpha_i X^i$. The cryptanalysis is as follows:

Step 1: When the polarisation is done with more than D variables, we will have 0, and when it is done with less than D variables, we will not find 0. So the value D is easy to find by a few polarisations.

Step 2: Moreover, the polarisation with exactly D variables depends only on the monomial $\alpha_d X_d$ (all the other monomials give 0), so from this polarisation, we will find a $*$ law (we find a $*$ law on d variables, and this gives of course also a $*$ law on 2 variables: $X'_1 * (X'_2 * ... * X'_n)$).

$f'(X) = \sum_{i=1}^{D} \beta_i X_i$ such that the public equations come from f'.

Step 4: Finally, a cleartext can be found from a ciphertext without knowing the secret affine functions s and t: we will just use f' and the $*$ law instead of f and the standard multiplication.

Remark: In the HFE schemes of [20] (with public polynomials of degree 2), the degree D of the hidden polynomial f is always larger than p because we want not only the monomials x and x^2 in f, so we must have at least one $x^{q^i+q^j}$ with $q^i + q^j > p$. So it seems that this "polarisation attack" does not work against the HFE schemes.

7 Another cryptanalysis of D^*

A few months after our first cryptanalysis (given in section 4), Nicolas Courtois found a very different cryptanalysis of D^*. We explain his cryptanalysis below.

Step 1: In the description of the D^* scheme, $s : K^n \mapsto \mathcal{L}_n$ is a *linear* and bijective application. Let us replace x by $x' = \sigma^{-1}(x)$ in the public equations of D^*, where $\sigma : K^n \mapsto K^n$ is an affine bijection, which is *not linear*. We obtain a new set of n public polynomials Q_1, ..., Q_n of total degree two in x'_1, ..., x'_n.
The first step of this attack consists in writing the affine applications $s \circ \sigma$ and t^{-1} as follows:

$$a = s \circ \sigma(x') = S_0 + \sum_{i=1}^{n} x'_i S_i,$$

$$b = t^{-1}(y) = \sum_{i=1}^{n} y_i T_i,$$

where S_0, S_1, ..., S_n and T_1, ..., T_n are elements of $\mathcal{L}_n$. Since s and t are bijective, $(S_i)_{1 \le i \le n}$ and $(T_i)_{1 \le i \le n}$ are two bases of $\mathcal{L}_n$. Moreover, since σ is not linear, we have $S_0 \neq 0$, and we can even suppose that $S_0 = 1$ (because for any $\lambda \in \mathcal{L}_n \setminus \{0\}$, we can change (s, t) into $(\frac{1}{\lambda} \cdot s, \lambda^2 \cdot t)$ and obtain the same cryptosystem.)
The D^* cryptosystem can thus be rewritten as follows:

$$\Big(1 + \sum_{i=1}^{n} x'_i S_i\Big)^2 = \sum_{i=1}^{n} y_i T_i.$$

Step 2: In the previous equation, if we replace each y_i by its public expression $Q_i(x'_1, ..., x'_n)$, we obtain an equation of total degree two in the x'_i variables, which holds for any x:

$$1 + 2\sum_{i=1}^{n} x'_i S_i + \sum_{i=1}^{n} {x'_i}^2 S_i^2 + 2 \sum_{1 \le i \ne j \le n} x'_i x'_j S_i S_j$$

$$= \sum_{i=1}^{n} \alpha_i T_i + \sum_{i=1}^{n} x'_i \Big(\sum_{j=1}^{n} \beta_{ij} T_j \Big) + \sum_{i=1}^{n} {x'_i}^2 \Big(\sum_{j=1}^{n} \gamma_{ij} T_j \Big) + \sum_{1 \le i \ne j \le n} x'_i x'_j \Big(\sum_{k=1}^{n} \delta_{ijk} T_k \Big),$$

where the coefficients α_i, β_{ij}, γ_{ij}, δ_{ijk} are known elements of K.
If we successively take $x = (0, ..., 0)$, $x = (1, 0, ..., 0)$, $x = (0, 1, 0, ..., 0)$, ..., $x = (0, ..., 0, 1)$, $x = (2, 0, ..., 0)$, $x = (0, 2, 0, ..., 0)$, ..., $x = (0, ..., 0, 2)$, $x = (1, 1, 0, ..., 0)$, $x = (1, 0, 1, 0, ..., 0)$, ..., $x =$

$$\begin{cases} 1 = \sum_{i=1}^{n} \alpha_i T_i & (1) \\ 2S_i = \sum_{j=1}^{n} \beta_{ij} T_j \quad (1 \le i \le n) & (2) \\ S_i^2 = \sum_{j=1}^{n} \gamma_{ij} T_j \quad (1 \le i \le n) & (3) \\ 2S_i S_j = \sum_{k=1}^{n} \delta_{ijk} T_k \quad (1 \le i \ne j \le n) & (4) \end{cases}$$

Step 3: Gaussian reductions on the equations (2) give the T_i as linear combinations of the S_j (the inversion is certainly possible, because $(S_i)_{1 \le i \le n}$ is a basis of $\mathcal{L}_n$):

$$\forall i,\ 1 \le i \le n,\ T_i = \sum_{j=1}^{n} \theta_{ij} S_j. \qquad (*)$$

In the same way, we also have, from (1) and $(*)$:

$$1 = \sum_{j=1}^{n} \varphi_j S_j. \qquad (**)$$

Step 4: By using the equations $(*)$ in (3) and (4), we obtain a "multiplication table" for the S_i:

$$S_i \cdot S_j = \sum_{k=1}^{n} \nu_{ijk} S_k \qquad (1 \le i, j \le n).$$

The complexity of this first part of the attack is $\mathcal{O}(n^3)$ (due to the gaussian reductions used in step 3 to obtain the T_i as linear combinations of the S_j).

Step 5: With the "multiplication table" above, it is possible to find a cleartext $x = (x_1, ..., x_n)$ from a given ciphertext $y = (y_1, ..., y_n)$, as follows. By definition,

$$s \circ \sigma(x') = \pm (t^{-1}(y))^{\frac{q^n+1}{4}}$$

if we let $x' = \sigma^{-1}(x)$. With the notations of this section, this gives:

$$1 + \sum_{i=1}^{n} x'_i S_i = \pm \Big(\sum_{i=1}^{n} y_i T_i \Big)^{\frac{q^n+1}{4}}$$

By using $(*)$, we obtain:

$$1 + \sum_{i=1}^{n} x'_i S_i = \pm \Big(\sum_{j=1}^{n} \eta_j S_j \Big)^{\frac{q^n+1}{4}}$$

Then, by using the "square and multiply" principle, together with the "multiplication table" of Step 5, and equation $(**)$, we obtain:

$$\sum_{i=1}^{n} x'_i S_i = \sum_{j=1}^{n} \psi_j S_j,$$

where the ψ_j are known coefficients of K (there are two possibilities for the ψ_j, because of the $\pm$ above). Since $(S_i)_{1 \le i \le n}$ is a basis of $\mathcal{L}_n$, we deduce: $(x'_1, ..., x'_n) = (\psi_1, ..., \psi_n)$. Finally, for each of the two solutions x', we compute $x = \sigma(x')$ (σ is known), and the cleartext is the solution x that satisfies $x \in \mathcal{M}$.

8 How to find s and t in the D^* cryptosystem ?

Following his attack of section 7, Nicolas Courtois has also found how to find the secret linear bijections s and t. We describe his method below. The notations are the same as in section 7.

of the (cyclic) multiplicative group of the algebraic extension $\mathcal{L}_n$. Actually, there are $\varphi(q^n-1)$ such generators, so that the probability of finding one is:

$$\frac{\varphi(q^n-1)}{q^n-1} = \prod_{p|q^n-1}\left(1-\frac{1}{p}\right) \geq \prod_{p\leq q^n}\left(1-\frac{1}{p}\right).$$

According to Mertens' formula, we have:

$$\prod_{p\leq N}\left(1-\frac{1}{p}\right) \sim \frac{e^{-\gamma}}{\log N},$$

where γ is Euler's constant. Therefore, we can obtain a generator in $\mathcal{O}(n\log q)$ tries in average.

Step 2: With the notations above, we can write:

$$a = 1 + \sum_{i=1}^{n} x'_i S_i,$$

where $x' = \sigma^{-1}(x)$.
By using $(**)$, the "square and multiply" principle, and the "multiplication table" of section 7, we can write:

$$a^k = \sum_{i=1}^{n} \zeta_{ki} S_i \qquad (0 \leq k \leq n),$$

where the ζ_{ki} are known coefficients of K that can be computed in polynomial time.

Step 3: Since $\mathcal{L}_n$ is a vector space of dimension n over K, the vectors 1, a, a^2, ..., a^n are linked, *i.e.* we can find coefficients ξ_0, ..., ξ_n of K such that:

$$\sum_{k=0}^{n} \xi_k a^k = 0,$$

i.e. $\Phi(a) = 0$, where $\Phi(X) = \sum_{k=0}^{n} \xi_k X^k$ is a polynomial of degree n with coefficients in K.

Step 4: We then compute all the roots of Φ in $\mathcal{L}_n$. For example, with the Berlekamp-Rabin algorithm, they can be found in $\mathcal{O}(n^3 \log n \log q)$ in average. We obtain at most n roots. It remains to find which one is the correct value of $a = s(x)$.

Step 5: Let a_0 be one of the roots found in Step 4. We can notice that the elements 1, a_0, ..., a_0^{n-1} of $\mathcal{L}_n$ are linearly independent over K. Actually, if they were not, there would be a polynomial Π of degree $\leq n-1$ such that $\Pi(a_0) = 0$. The group generated by a_0 in $\mathcal{L}_n$ would therefore be contained in the set:

$$\{\rho(a_0),\ \rho \in K[X] \setminus \{0\},\ d^o\rho \leq n-2\},$$

whose cardinality is $q^{n-1} - 1 < q^n - 1$. As a result, a_0 could not be a generator of the multiplicative group of $\mathcal{L}_n$, a contradiction.
We thus obtain a system of n linearly independent equations on S_1, ..., S_n:

$$\forall k,\ 0 \leq k \leq n-1,\ \sum_{i=1}^{n} \zeta_{ki} S_i = a_0^k.$$

This system has a unique solution $(S_1, ..., S_n)$.
When S is known, T can be deduced with equations $(*)$, and we can check if those S and T are correct.

We can repeat Step 5 with each root of Φ, until we find the correct one: that gives an algorithm in $\mathcal{O}(n^4 \log n \log q)$ to find S and T.

that $\tilde{\Phi}(a) = 0$ (by considering other powers of a in Step 3) and computing $\gcd(\Phi, \tilde{\Phi})$, whose degree is low with a high probability. The computation of the correct value of a is then likely to be much easier (we can expect the degree of $\gcd(\Phi, \tilde{\Phi})$ to be one with a rather high probability).

Part III
Our new candidate

9 The D^{**} algorithm

As we saw in the previous part, an encryption scheme based on the use of a monomial transformation of small degree (more precisely, of smaller degree than the characteristic of the field K) is insecure. A natural idea is then to design a cryptosystem that uses *two* rounds of D^*-like transformations.

9.1 Representation of the message

We choose the same field K and the same message space $\mathcal{M}$ as in the description of D^* (see section 3).

9.2 Encryption of $x \in \mathcal{M}$

The scheme also makes use of an extension $\mathcal{L}_n$ of degree n over K (which can be fixed, as we mentioned before), and three linear secret bijections $s : K^n \to \mathcal{L}_n$, $t : \mathcal{L}_n \to \mathcal{L}_n$ and $u : \mathcal{L}_n \to K^n$ (each of them can be given by n polynomials in n variables over K, and of total degree one).

With these notations, the ciphering algorithm can be described as follows. $y = H(x)$ is defined as the only element of $\{ + u(t(s(x)^2)^2), -u(t(s(x)^2)^2)\}$ that belongs to $\mathcal{M}$ (this element exists and is unique, by construction of $\mathcal{M}$). Since s, t and u are of total degree one over K, $u(t(s(x)^2)^2)$ can be given in a basis by n polynomials P_1, ..., P_n of total degree 4 in n variables, whose coefficients belong to K. These polynomials are made public, so that anyone can easily encrypt a message, by using the following equations to compute $y = (y_1, ..., y_n) = H(x)$ from $x = (x_1, ..., x_n) \in \mathcal{M}$:

$$\begin{cases} y = (y_1, ..., y_n) \in \mathcal{M} \\ y_1 = \pm P_1(x_1, ..., x_n) \\ ... \\ y_n = \pm P_n(x_1, ..., x_n) \end{cases}$$

9.3 Decryption of $y \in \mathcal{M}$

As for D^*, H is a permutation from $\mathcal{M}$ to $\mathcal{M}$, and the decryption of $y \in \mathcal{M}$ is also very easy, when the secret linear permutations s, t and u are known. For any $y \in \mathcal{M}$, $x = H^{-1}(y)$ is given by the following formula:

$$\begin{cases} x \in \mathcal{M} \\ x = \pm s^{-1}\Big((t^{-1}(u^{-1}(y)))^{\frac{q^n+1}{4}})^{\frac{q^n+1}{4}}\Big). \end{cases}$$

10 Complexity of functional decomposition

In this section, we consider a natural attack on the D^{**} scheme. This attack consists in trying to "separate" the two rounds of D^{**}. This leads to the following problem:

Decomposition problem: Let g and h be two functions which map K^n into K^n and which are given by polynomials of total degree two in n variables over K. Then $f = g \circ h$ is also a function from K^n to K^n, and it is given by n polynomials of degree *four* in n variables over K. Suppose that f is given. Is it computationally feasible to recover g and h ?

from each other. As a result, to break the scheme, we would only have to break two independent D^* schemes, and that is feasible as we saw in section 4.
That would of course make the general idea of using two rounds uninteresting. However, the decomposition of multivariate polynomials was studied by Matthew Dickerson, who gave an algorithm for the following problem:

Multivariate left decomposition: Given polynomials f and h_1,...,h_n in $K[X_1, ..., X_n]$, and an integer r, decide if there exists a polynomial $g(x_1, ..., x_n)$ of total degree at most r that composes with the h_i's to give f. That is, does there exist a polynomial $g(x_1, ..., x_n)$ such that

$$f(x_1, ..., x_n) = g(h_1(x_1, ..., x_n), ..., h_n(x_1, ..., x_n))$$

and deg$(g) \leq r$? If so, determine the coefficients of g.

In [4], Dickerson presents the best-known algorithm for this problem, which is polynomial in the degree of $f, h_1, ..., h_n$, but *exponential* in the number n of variables (note that our decomposition problem is even harder, because the h_i's are not known).
He also shows that the *general* problem of decomposition of multivariate polynomials is difficult, because the following one is NP-hard:

s-1-decomposition problem: Given a monic univariate polynomial $f(x)$ and an integer s, decide if there exists an s-1-decomposition of f, *i.e.* a monic univariate polynomial h of degree s, and a bivariate polynomial $g(y, x) \in K[Y, X]$ of the form $g(y, x) = \prod_{i=1}^{r}(y + \alpha_i x + \beta_i)$ with $\alpha_i, \beta_i \in \hat{K}$, an algebraic extension of K, such that $f(x) = g(h(x), x)$. If so, determine the coefficients of g and h.

There are some reasons to think that the following problem is also NP-hard (cf [4], problem 14, p. 74):

Multivariate decomposition of given degree: Given a polynomial f in $K[X_1, ..., X_n]$ and some subset of the following: integers $k, r, s_1, ..., s_k$, a polynomial $g(x_1, ..., x_k) \in K[X_1, ..., X_k]$, and polynomials $h_1(x_1, .., x_n)$,..., $h_k(x_1, ..., x_n)$, decide if there exists a functional decomposition $g, h_1, ..., h_k$ of f such that deg$g = r$, and deg$h_i = s_i$ for $1 \leq i \leq n$. If so, compute those coefficients of g and the h_i's which were not given.

Dickerson (see [4], p. 75) notices that: "The s-1-decomposition problem seems intuitively easier than problem 14. In problem 14, f, g and h are general multivariate polynomials of arbitrary dimension. Furthermore polynomial g takes the polynomial h_i as arguments, and we know nothing about the form of g other than its degree. In the s-1-decomposition problem, on the other hand, f and h are both univariate polynomials and g is only bivariate. Furthermore, g takes x and not another polynomial as its second argument. We also know a great deal about the structure of the polynomial g, namely that it factors as: $g(y, x) = \prod_{i=1}^{r}(y + \alpha_i x + \beta_i)$. However, we have tried without success to reduce the s-1-decomposition problem to problem 14."
Therefore, it is still an open problem... and – at the present – it does not lead to any practical attack on D^{**}.

11 Comparison with RSA in secret key computations

The aim of this section is to compare the speed of a realistic implementation of D^{**} with the speed of the standard 512 bits RSA cryptosystem.
We take $p = q = 251$, and $n = 9$, so that each message is about 72 bits large. By a careful study of the exponentiation $b \mapsto b^{\frac{q^n+1}{4}}$, it can be proved that D^{**} – in this example – requires less than 50 multiplications over $\mathcal{L}_n$ in secret key computations.
We can therefore summarize the complexity of secret key computations as follows:

$$\begin{cases} \leq 50 \text{ multiplications } 72 \text{ bits} \times 72 \text{ bits} & \text{for } D^{**} \\ \simeq 768 \text{ multiplications } 512 \text{ bits} \times 512 \text{ bits} & \text{for RSA.} \end{cases}$$

As a result, the secret key computations in D^{**} are expected to be at least 100 times faster than those of RSA.

The D^{**} algorithm is built with two rounds of D^* algorithms. We can also design a variation of this D^{**} scheme, called TD^*, where the first round will be a "triangular" or "mixed triangular/D^*" scheme, and where the second round is still a D^*. By "triangular", we mean a transformation T of the following type:

$$T(a_1, \ldots, a_n) = (a_1^2, a_2^2 + q_2(a_1), a_3^2 + q_3(a_1, a_2), \ldots, a_n^2 + q_n(a_1, \ldots, a_{n-1})),$$

where $q_2, \ldots, q_n$ are homogeneous polynomials of total degree two.
By "mixed triangular/D^*" scheme, we mean a transformation f such that $f(A||B) = D^*(A)||T(B) + P(A)$, where $||$ is the concatenation function, where T is a "triangular" scheme, and where P is a homogeneous polynomial of total degree two. These TD^* schemes are also candidate trapdoor one-way permutations.

Note: The "triangular" or "mixed triangular/D^*" function must be in the first round. If the two rounds are put the other way round, the scheme can easily be broken.

13 Attacks agains 2R schemes

In 1999, in [5],an algorithm that is often able to find the decomposition of two quadratic multivariate polynomials has been published. This algorithm is expected to break the 2R schemes (among them D^{**}) when all the composition is given in the public key. In order to repair the 2R schemes (for example D^{**}), we can suggest:

- To not publish all the originally public equations (it gives a $2R^-$ scheme).
- Or to introduce a "perturbation" on these equations, for example by introducing some extra variables (it gives a 2RV scheme), by fixing some variables (ir gives a 2RF scheme), or by mixing the equations cith truly random equations (it gives a $2R^+$ scheme). Moreover, all these "perturbations" can be combined (it gives a $2R^{+-}$VF scheme). See [22] for more details (in [22], these "perturbations" are used and studied on the C^* scheme).

Note: When these perturbations are done on D^{**} or TD^*, no attacks are known against the resulting schemes (D^{**-}, D^{**+}, $D^{**}V$; $D^{**}F$, etc). However, unlike D^{**} or TD^*, these schemes are not bijective anymore...

14 Conclusion

From any trapdoor one-way permutation, it is easy to build asymmetric schemes for ciphering, signature, or authentication. However, very few candidate trapdoor one-way permutations are known at the present. Therefore, we think that all the candidates should be studied carefully, and that one should go on looking for new candidates.

In this paper, we have quickly described all the know candidates we are aware of. They can be split into two families: the "RSA-like" family and the "multivariate polynomial" family. The "RSA-like" family contains all the schemes that can be seen as generalizations of the RSA scheme (Rabin-Williams, Dickson, Elliptic curves analogues of RSA). In the "multivariate polynomial" family, the public key is given as a set of multivariate polynomials. The original C^* scheme of T. Matsumoto and H. Imai was a typical example of this family, but it is known to be insecure. However, it is possible to design some other schemes in this family, such as the B scheme [13], or such as the new schemes D^{**} and TD^* described in this paper.

Of course, the candidates of the "RSA-like" family look more "serious" than others. No candidates are known with an absolute proof of security, but the security of these candidates is related to a famous open problem, such as factoring, or computing e-th roots modulo n, whereas the security of the other candidates is just an open problem, not related to a famous problem.

published in [5]. However, it is very easy to repair the schemes, *i.e.* to avoid the cryptanalysis of [5], for example by not publishing all the originally public equations. The resulting schemes, called $2R^-$, are not broken (but they are not bijective anymore...). For example for D^{**-} or TD^{*-}, no efficient attacks are known, and we keep the property that secret key computations are very easy. Secret key computations of D^{**-} or TD^{*-} are more than 100 times faster than RSA and they can be performed in low-cost smartcards (*i.e.* without co-processors). We hope that this paper will support the arising of new ideas in asymmetric cryptography.

References

[1] Daniel Bleichenbacher, Wieb Bosma, Arjen K. Lenstra, *Some Remarks on Lucas-Based Cryptosystems*, Advances in Cryptology, Proceedings of CRYPTO'95, Springer, pp. 386-396.

[2] Ian Blake, XuHong Gao, Ronald Mullin, Scott Vanstone, Tomik Yaghoobian, *Applications of finite Fields*, Kluwer Academic Publishers, p. 25.

[3] N. Demytko, *A New Elliptic Curve Based Analogue of RSA*, Advances in Cryptology, Proceedings of EUROCRYPT'93, Springer-Verlag, pp. 40-49.

[4] Matthew Dickerson, *The functional Decomposition of Polynomials*, Ph.D Thesis, TR 89-1023, Department of Computer Science, Cornell University, Ithaca, NY, July 1989.

[5] Y. Ding-Feng, L. Kwok-Yan, D. Zong-Duo, *Cryptanalysis of "2R" Schemes*, Proceedings of CRYPTO'99, Springer, pp. 315-325.

[6] Michael Garey, David Johnson, *Computers and Intractability, a Guide to the Theory of NP-Completeness*, Freeman, p. 251.

[7] Guang Gong, Lein Harn, *Public-Key Cryptosystems Based on Cubic Finite Field Extensions*, to appear in IEEE Transactions on Information Theory.

[8] Kenji Koyama, Ueli M. Maurer, Tatsuaki Okamoto, Scott A. Vanstone, *New Public-Key Schemes Based on Elliptic Curves over the Ring* $\mathbf{Z}_n$, Advances in Cryptology, Proceedings of CRYPTO'91, Springer-Verlag, pp. 252-266.

[9] Kaoru Kurosawa, Toshiya Itoh, Masashi Takeuchi, *Public key cryptosystem using a reciprocal number with the same intractactability as factoring a large number*, Cryptologia, vol. 12, n° 4, pp. 225-233, 1988.

[10] John H. Loxton, David S.P. Khoo, Gregory J. Bird, Jennifer Seberry, *A Cubic RSA Code Equivalent to Factorization*, Journal of Cryptology, v.5, n.2, 1992, pp. 139-150.

[11] Rudolf Lidl, G.L. Mullen, G. Turwald, *Pitman Monographs and Surveys in Pure and Applied Mathematics 65: Dickson Polynomials*, London, Longman Scientific and Technical, 1993.

[12] Rudolf Lidl, Winfried B. Müller, *Permutation Polynomials in RSA-Cryptosystems*, Advances in Cryptology, Proceedings of CRYPTO'83, Plenum Press, 1984, pp. 293-301.

[13] Tsutomu Matsumoto, Hideki Imai, *Algebraic Methods for Constructing Asymmetric Cryptosystems*, AAECC-3, Grenoble, 1985.

[14] Tsutomu Matsumoto, Hideki Imai, *Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption*, Advances in Cryptology, Proceedings of EUROCRYPT'88, Springer-Verlag, pp. 419-453.

[15] Winfried B. Müller, *Polynomial Functions in Modern Cryptology*, Contributions to General Algebra 3: Proceedings of the Vienna Conference, Vienna: Verlag Hölder-Pichler-Tempsky, 1985, pp. 7-32.

tiarum Mathematicarum Hungarica, v.16, 1981, pp. 71-76.

[17] Winfried B. Müller, Rupert Nöbauer, *Cryptanalysis of the Dickson-scheme*, Advances in Cryptology, Proceedings of EUROCRYPT'85, Springer-Verlag, pp. 50-61.

[18] Pascal Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, Advances in Cryptology, Proceedings of EUROCRYPT'99, Springer, pp. 223-238.

[19] Jacques Patarin, *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88*, Advances in Cryptology, Proceedings of CRYPTO'95, Springer, pp. 248-261.

[20] Jacques 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.

[21] Jacques Patarin, *Asymmetric Cryptography with a Hidden Monomial*, Advances in Cryptology, Proceedings of CRYPTO'96, Springer, pp. 45-60.

[22] Jacques Patarin, Louis Goubin, Nicolas Courtois, C^*_{-+} *and HM: Variations around two schemes of T. Matsumoto and H. Imai*, Advances in Cryptology, Proceedings of ASIACRYPT'98, Springer, pp. 45-60.

[23] M.O. Rabin, *Digitized Signatures and Public-Key Functions as Intractable as Factorization*, Technical Report LCS/TR-212, M.I.T. Laboratory for Computer Science, 1979.

[24] R.L. Rivest, A. Shamir, L.M. Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM, v.21, n.2, 1978, pp. 120-126.

[25] P.J. Smith, *LUC Public-Key Encryption*, Dr. Dobb's Journal, January 1993, pp. 44-49.

[26] P.J. Smith, M.J.J. Lennon, *LUC: a New Public Key System*, Proceedings of the Ninth IFIP Int. Symp. on Computer Security, 1993, pp. 103-117.

[27] H.C. Williams, *A Modification of the RSA Public-Key Encryption Procedure*, IEEE Transactions on Information Theory, v.IT-26, n.6, 1980, pp. 726-729.

[28] H.C. Williams, *An* M^3 *Public-Key Encryption Scheme*, Advances in Cryptology, Proceedings of CRYPTO'85, Springer-Verlag, pp. 358-368.

Solving a system of quadratic equations over any field is NP-complete

It is known that solving a randomly selected system of multivariate quadratic equations over the field $K = GF(2)$ is an NP-complete problem (see [6] page 251). In this appendix, we show that this is still the case when K is *any* field. Moreover, as concerns the case $K = GF(2)$, Garey and Johnson refer to a paper that was never published, so that we give a proof for this case too.

1. The case $K = GF(2)$

Let us consider an instance of the 3-Satisfiability problem (also called 3-SAT), given by a finite set $U = \{u_1, ..., u_n\}$ of Boolean variables, and a collection $C = \{c_1, ..., c_m\}$ of clauses on X. By definition, each clause is a disjonction of at most three litterals over U (a litteral is some u or some $\bar{u}$, with $u \in U$). Each Boolean variable can be considered in an obvious way as an element of $GF(2)$. Moreover:

- If $u \in U$ corresponds to $x \in GF(2)$, then $\bar{u}$ correponds to $1 - x$.
- If $u \in U$ (resp. $v \in U$) corresponds to $x \in GF(2)$ (resp. $y \in GF(2)$), then (u or v) corresponds to $(xy + x + y)$ in $GF(2)$.

Therefore, finding a truth assigment for U that satisfies all the clauses in C is equivalent to solving some system of m *cubic* equations in n indeterminates over $GF(2)$. Moreover, each equation of this system contains at most three of the x_i indeterminates. If we add the $\frac{n(n-1)}{2}$ new variables $z_{ij} = x_i x_j$ $(i < j)$, the system can be rewritten into a system of $m + \frac{n(n-1)}{2}$ *quadratic* equations in $\frac{n(n+1)}{2}$ indeterminates over $GF(2)$.

Conclusion The problem of solving a randomly selected instance of the 3-SAT problem – which is NP-complete – can be reduced in polynomial time to the problem of solving a system of randomly selected multivariate quadratic equations over the $GF(2)$. As a result, this latter problem is also NP-complete.

2. The general case

Let K be *any* field, and let $\mathcal{S}$ be the following system of n quadratic equations in n indeterminates over $GF(2)$:

$$\sum_{1 \leq i < j \leq n} \mu_{ijk} x_i x_j + \sum_{i=1}^{n} \nu_{ik} x_i = y_k \qquad (1 \leq k \leq n),$$

where the y_k are fixed elements of $GF(2)$. We are going to show that the problem of solving the system $\mathcal{S}$ can be reduced – in polynomial time – to the problem of solving some system of quadratic equations over K.

Transformation of the system Over $GF(2)$, solving $(\mathcal{S})$ is equivalent to solving the following system:

$$(\mathcal{S}') \begin{cases} \mu_{12k} x_1 x_2 = z_{12k} & (1 \leq k \leq n) \\ \mu_{13k} x_1 x_3 = z_{12k} + z_{13k} & (1 \leq k \leq n) \\ \quad \text{.....} & \\ \mu_{(n-1)nk} x_{n-1} x_n = z_{(n-2)nk} + z_{(n-1)nk} & (1 \leq k \leq n) \\ \nu_{1k} x_1 = z_{(n-1)nk} + w_{1k} & (1 \leq k \leq n) \\ \nu_{2k} x_2 = w_{1k} + w_{2k} & (1 \leq k \leq n) \\ \quad \text{.....} & \\ \nu_{(n-1)k} x_{n-1} = w_{(n-2)k} + w_{(n-1)k} & (1 \leq k \leq n) \\ \nu_{nk} x_n = w_{(n-1)k} + y_k & (1 \leq k \leq n) \end{cases}$$

It is a system of $\frac{n^2(n+1)}{2}$ equations over $GF(2)$, with $\frac{n^2(n+1)}{2}$ indeterminates: the x_i $(1 \leq i \leq n)$, the z_{ijk} $(1 \leq i < j \leq n,\ 1 \leq k \leq n)$, and the w_{ik} $(1 \leq i \leq n-1,\ 1 \leq k \leq n)$.

(where 0 denotes the identity element of the addition law of K, et 1 denotes the identity element of the multiplication law of K) such that $x(x-1)=0$. The multiplication law $(x,y)\mapsto xy$ of $GF(2)$ can be translated into $(x,y)\mapsto xy$ over K, whereas one can translate the addition law $(x,y)\mapsto x+y$ of $GF(2)$ into $(x,y)\mapsto (x+y)(2-(x+y))$ over K (where 2 denotes the element $1+1$ of K).
As a result, solving $(\mathcal{S}')$ over $GF(2)$ is equivalent to solving the following system over K:

$$(\mathcal{S}'')\begin{cases}
\mu_{12k}x_1x_2 = z_{12k} & (1\leq k\leq n)\\
\mu_{13k}x_1x_3 = (z_{12k}+z_{13k})(2-z_{12k}-z_{13k}) & (1\leq k\leq n)\\
\quad \ldots\ldots\ldots\ldots & \\
\mu_{(n-1)nk}x_{n-1}x_n = (z_{(n-2)nk}+z_{(n-1)nk})(2-z_{(n-2)nk}-z_{(n-1)nk}) & (1\leq k\leq n)\\
\nu_{1k}x_1 = (z_{(n-1)nk}+w_{1k})(2-z_{(n-1)nk}-w_{1k}) & (1\leq k\leq n)\\
\nu_{2k}x_2 = (w_{1k}+w_{2k})(2-w_{1k}-w_{2k}) & (1\leq k\leq n)\\
\quad \ldots\ldots\ldots\ldots & \\
\nu_{(n-1)k}x_{n-1} = (w_{(n-2)k}+w_{(n-1)k})(2-w_{(n-2)k}-w_{(n-1)k}) & (1\leq k\leq n)\\
\nu_{nk}x_n = (w_{(n-1)k}+y_k)(2-w_{(n-1)k}-y_k) & (1\leq k\leq n)\\
x_i(x_i-1)=0 & (1\leq i\leq n)\\
z_{ijk}(z_{ijk}-1)=0 & (1\leq i<j\leq n,\ 1\leq k\leq n)\\
w_{ik}(w_{ik}-1)=0 & (1\leq i\leq n-1,\ 1\leq k\leq n)
\end{cases}$$

It is a system of $n^2(n+1)$ quadratic equations with $\frac{n^2(n+1)}{2}$ indeterminates over K.

Conclusion The problem of solving a randomly selected system of multivariate quadratic equations over $GF(2)$ – which is NP-complete – can be reduced in polynomial time to the problem of solving a system of randomly selected multivariate quadratic equations over the field K. Therefore, this latter problem is also NP-complete. Note that K can even be a division ring, because the proof does not use the commutativity of the multiplication law in K.