An Application of Algebraic Geometry to Encryption Tame Transformation Method

T. Moh*

May 31, 2001, revised Dec 3, 2002

dedicated to Professor Jose Luis Vicente Cordoba for his Sixty's Birthday

Abstract

Let $\mathbf{K}$ be a finite field of 2^ℓ elements. Let $\phi_4, \phi_3, \phi_2, \phi_1$ be *tame* mappings of the $n+r$-dimensional affine space $\mathbf{K}^{n+r}$. Let the composition $\phi_4\phi_3\phi_2\phi_1$ be π. The mapping π and the ϕ_i's will be hidden. Let the component expression of π be $(\pi_1(x_1, \cdots, x_{n+r}), \cdots, \pi_{n+r}(x_1, \cdots, x_{n+r}))$. Let the restriction of π to a subspace be $\hat{\pi}$ as $\hat{\pi} = (\pi_1(x_1, \cdots, x_n, \text{n}0, \cdots, 0), \cdots, \pi_{n+r}(x_1, \cdots, x_n, 0, \cdots, 0)) = (f_1, \cdots, f_{n+r}) : \mathbf{K}^n \mapsto \mathbf{K}^{n+r}$. The field $\mathbf{K}$ and the polynomial map $(f_1, \cdots, f_{n+r})$ will be announced as the public key. Given a plaintext $(x_1', \cdots, x_n') \in \mathbf{K}^n$, let $y_i' = f_i(x_1', \cdots, x_n')$, then the ciphertext will be $(y_1', \cdots, y_{n+r}') \in \mathbf{K}^{n+r}$. Given ϕ_i and $(y_1', \cdots, y_{n+r}')$, it is easy to find $\phi_i^{-1}(y_1', \cdots, y_{n+r}')$. Therefore the plaintext can be recovered by $(x_1', \ldots, x_n', 0, \cdots, 0) = \phi_1^{-1}\phi_2^{-1}\phi_3^{-1}\phi_4^{-1}\hat{\pi}(x_1', \cdots, x_n') = \phi_1^{-1}\phi_2^{-1}\phi_3^{-1}\phi_4^{-1}(y_1', \cdots, y_{n+r}')$. The private key will be the set of maps $\{\phi_1, \phi_2, \phi_3, \phi_4\}$. The security of the system rests in part on the difficulty of finding the map π from the partial informations provided by the map $\hat{\pi}$ and the factorization of the map π into a product (i.e., composition) of tame transformations ϕ_i's.

keywords: tame transformation, public key system, public key, private key, plaintext, ciphertext, signature, master key, error-detect.

1 Introduction

There is a long history of studying 'automorphism groups' for affine spaces $\mathbf{K}^n$ and 'embedding theory' in mathematics. There are thousands of papers on those subjects. There is a simple concept of *Tame* transformations, which is defined as,

Definition: We define a *tame* transformation $\phi_i = (\phi_{i,1}, \cdots, \phi_{i,m})$ as either a linear transformation, or of the following form in any *order* of variables $x_1, \cdots, x_n$ with polynomials $h_{i,j}$,

$$(1) : \phi_{i,1}(x_1, \cdots, x_n) = x_1 + h_{i,1}(x_2, \cdots, x_n) = y_1$$

*Math Department, Purdue University, West Lafayette, Indiana 47907-1395. tel: (765)-494-1930, e-mail ttm@math.purdue.edu

$$(2) : \phi_{i,2}(x_1, \cdots, x_n) = x_2 + h_{i,2}(x_3, \cdots, x_n) = y_2$$
$$\cdots\cdots\cdots$$
$$(j) : \phi_{i,j}(x_1, \cdots, x_n) = x_j + h_{i,j}(x_{j+1}, \cdots, x_n) = y_j$$
$$\cdots\cdots\cdots$$
$$(n) : \phi_{i,n}(x_1, \cdots, x_n) = x_n = y_n$$

If the tame transformation is invertible, then it is called a tame automorphism. ■

It is not hard to see that the inverse of a tame automorphism of $\mathbf{K}^n$ is a tame automorphism, and the subgroup generated by all tame automorphisms is called the tame automorphism group, Tame($\mathbf{K}^n$). Therefore, for any affine space $\mathbf{K}^n$, there are two groups, the automorphism group Auto($\mathbf{K}^n$) and Tame($\mathbf{K}^n$). For the case that $n = 2$, the beautiful theory of W. Van der Kulk in 1953 in [27] states that Auto($\mathbf{K}^2$)=Tame($\mathbf{K}^2$), i.e, any automorphism of $\mathbf{K}^2$ can be written as a canonical product of tame automorphisms.

There is an abyss between our knowledge of the automorphism group of $\mathbf{K}^2$ and the automorphism group of $\mathbf{K}^n$ for $n \geq 3$. Can the theory of W. Van der Kulk be generalized to higher dimensional cases? We do not know the answer. Even worse, we do not have a factorization theorem for Tame($\mathbf{K}^n$) if $n \geq 3$, i.e., for $n \geq 3$, every element π in the tame automorphism group has a factorization $\pi = \prod_i \phi_i$ by its definition, however, there is no known way to find the factorization. In [22], Nagata constructed an automorphism as follows; $\sigma(x_1, x_2, x_3) = (x_1, x_2 + x_1(x_1x_3 + x_2^2), x_3 - x_2(x_1x_3 + x_2^2) - x_1(x_1x_3 + x_2^2)^2)$ for $n = 3$ and raised the question that if it is in the tame automorphism group. Note that if we have a factorization theorem for element in the tame automorphism group, then one may simply assume that the above element is in the tame automorphism group, and do the factorization, if one succeeds, then naturally it is in the tame automorphism group, otherwise not. We can not answer Nagata's question after some forty years, simply because we do not know how to factor elements in the tame automorphism group.

For embedding theory ([1], [18], [22]), the simplest case, i.e., the (algebraic) embedding of affine line to affine plane in characteristic 0, had been an open problem for forty years. It was solved in [1] using difficult and long arguments. The result is that any embedding mapping is a composition of a trivial mapping of the affine line to x-axis and an element of the tame automorphism group, or we should say that any embedding mapping is a tame transformation. It is unknown about how to generalize the above statement to either higher dimensional cases (i.e., affine lines to affine spaces or affine planes to affine spaces, etc.) in characteristic zero or even affine lines to affine planes in positive characteristics. There are some conjectures and discussions about the later cases in [18].

The above is the Algebraic Geometry background of this talk. We should discuss the background of Public-key Encryption theory in the below.

Let $\mathbf{A}$ be a set of data (or plaintexts, or signals) which one want to be written as another set $\mathbf{B}$ of data (or ciphertexts, or scrambled signals) to protect privacy or secrets. Mathematically, we have two sets $\mathbf{A}$ and $\mathbf{B}$ and a mapping π from $\mathbf{A}$ to $\mathbf{B}$. The mapping π is traditionally given by a "code book", maybe called as the "code" (nowaday, code largely means "self-correcting code", which is very different from what we are discussing. We shall stick to the term "encryption") or "key". In 1970's, motivated by high-speed computers, a new theory of encryption, public-key encryption, came to exist, The thinking is as follows. Although, given any one-one mapping π, its inverse mapping π^{-1} is uniquely determined mathematically. However, maybe computationally, the inverse mapping can not be found in a reasonable time (say, 10^{10} years) using the most powerful computer without the help of some extra informations (i.e., the "trap-door"). Then one may announce the mapping π (the "public-key) while keep the trap-door (the "private-key") secret. These are the essences of the public-key encryption theory.

We shall use the example of RSA encryption system to illustrate the point of public-key system. The mathematical foundation of it is some facts in the elementary number theory. Let p, q be two prime numbers close to 2^{512} (which are easy to be found), $n = pq$, d, e two positive integers such that

$$de = 1 \bmod (p-1)(q-1)$$

For any positive integer $x < n$, let

$$y = x^e \text{mod } n \text{ and } y < n$$

It is not hard to see that

$$x = y^d \text{mod } n$$

For encryption, we let $\mathbf{A}$=$\{x\}$, $\mathbf{B}$=$\{y\}$ and π the mapping defined by raising to the power of e and then mod n. The inverse mapping is given by raising to the power of d and then mod n. For practical use, let us consider the encryption of an e-mail. Every letter is giving by an eight bits number, we may glue 128 symbols together to form a number of 1024 bits, thus we chop the whole e-mail into many blocks of 1024 bits, and we treat each 1024 block as an element in $\mathbf{A}$, and encrypt it to an element in $\mathbf{B}$, and then send the encrypted e-mail. In the RSA system, the number n is announced with the public key e, while the private key d will be kept secret. The security of this system rely on the computationally difficulty in finding the private key from the public key e and the number n.

The other interesting system is the ElGamal system which is tied to the problems of finite commutative groups (as represented in the number theory); Find x, if exists, in the following equation $g^x = y$ where g is a given element of the group and y is an arbitrary element of the group ("discrete log problem"). This problem is surprisingly difficult computationally.

Let us consider the RSA system. To make the theory working, one has to glue symbols together to form large numbers, and thus slow down the arithmetical operations. To increase the security (i.e., to make it difficult to find the number d from the number e), one has to increase the size of n. Since we have

$$de = 1 \bmod (p-1)(q-1)$$

then most likely $d > n$. Since the ciphertext y has to be raised to the power of d, one may image that the deciphering process being slow.

Note that in the past the most successful public-key encryption systems, as RSA and ElGamal systems, are one dimensional. Their speeds might have to be accelerated by using hardwares, and their applications become expensive. From mathematical point of view, it will be nature to try higher dimension methods, i.e., multivariate public-key encryption systems. In Imai-Matsumoto theory ([14]), a polynomial of one variable (i.e., one dimensional) is expressed with respect to a field basis to achieve an expression of several variables (i.e., higher dimensional). Their attempt is noble, however, unsuccessful, since it had been broken by J. Patarin (cf [28]). Note that J. Patarin had proposed several extensions and generalizations of Imai-Matsumoto scheme which have not been broken.

We may use Algebraic Geometry of higher dimension to produce a public-key encryption system. For this purpose let us consider $a_1, ..., a_{128}$ being 128 characters with each an 8 bits number. The natural way is not to glue them together. We shall treat them as a point $a = (a_1, ..., a_{128})$ (a so called "plaintext") in 128 dimensional space. To scramble it, we simple apply a map ϕ to the 128 dimensional space and get a new point $b = (b_1, ..., b_{128})$ (a so called "ciphertext"). For the convenience of computation, we require that

(a) the maps π are non-linear to prevent an attack using linear algebra.

(b) both value $\pi(a)$ and its inverse value $\pi^{-1}(b)$ can be computed easily.

(c) each map is a composition of "simple" maps (in the sense of item (b) above) and shall be hard to be decompsed and its inverse hard to be recovered.

(d) it should be user-friendly.

In this paper we will show that the higher dimensional affine spaces, their "Tame Automorphism Group" (the group generated by all tame automorphisms) are tailored for this purposes.

2 Mathematical background

Since a mathematical theory, the theory of Tame Automorphisms, is applied to provide a public key system. We shall explain every term used in this lecture.

(a): Finite Field.

We shall discuss the concept of finite fields. The finite field $GF(2^m)$ of 2^m elements is the collection of the m bits numbers $(a_1, a_2, ..., a_m)$, where a_i's are zeroes or ones, and the sum of m bits numbers is bitwise, while the product depends on the defining irreducible polynomial, which can be carried out by a LSR (linear shift register) or by looking up a table.

(b): Affine Space.

Let K be a field, say $GF(2^m)$. Let K^{n+r} be the affine space of dimension n+r over K. Note that an "affine space" K^{n+r} is a vector space without the algebraic structure and the origin, i.e., the "physical space". We prefer an affine space over a vector space because (1) we need to remove the origin, (2) we shall consider non-linear maps such as polynomial maps.

(c): Tame Transformation.

A linear transformation $\psi = (\psi_1, ..., \psi_{n+r})$ is a map of the following form,

$$\psi_i(x_1, ..., x_{n+r}) = \sum_j a_{ij} x_j + b_i$$

where a_{ij} and b_i are elements in K. A linear transformation ψ is said to be *invertible* if the coefficient matrix (a_{ij}) is invertible.

Definition: We define a *tame* transformation $\phi_i = (\phi_{i,1}, \cdots, \phi_{i,n+r})$ as either a linear transformation, or of the following form in any *order* of variables $x_1, \cdots, x_{n+r}$ with polynomials $h_{i,j}$,

$$
\begin{aligned}
&(1) : \phi_{i,1}(x_1, \cdots, x_{n+r}) = x_1 + h_{i,1}(x_2, \cdots, x_{n+r}) = y_1 \\
&(2) : \phi_{i,2}(x_1, \cdots, x_{n+r}) = x_2 + h_{i,2}(x_3, \cdots, x_{n+r}) = y_2 \\
&\cdots\cdots\cdots \\
&(j) : \phi_{i,j}(x_1, \cdots, x_{n+r}) = x_j + h_{i,j}(x_{j+1}, \cdots, x_{n+r}) = y_j \\
&\cdots\cdots\cdots \\
&(n+r) : \phi_{i,n+r}(x_1, \cdots, x_{n+r}) = x_{n+r} = y_{n+r}
\end{aligned}
$$

If the tame transformation is invertible, then it is called a tame automorphism.

Example:

Let $K = GF(2), \psi(x_1, x_2, x_3) = (x_1 + x_2x_3, x_2 + x_3^2, x_3), \eta(x_1, x_2, x_3) = (x_1, x_2, x_3 + x_1^2)$ be two tame automorphisms. Then it is easy to see that $\psi^2(x_1, x_2, x_3) = (x_1 + x_3^3, x_2, x_3)$ and $\psi\eta(x_1, x_2, x_3) = (x_1 + x_2x_3 + x_1^2x_2, x_2 + x_3^2 + x_1^4, x_3 + x_1^2)$

The group generated by all tame automorphisms is called the *tame automorphism group.* Note that the group product is the composition of maps, i.e., **substitution**, which is different from the product of polynomials. The following proposition and its corollaries will be given without proofs.

Proposition 1 *Let a nonlinear tame transformation ϕ_i be defined as in the preceding paragraph. We have the inverse $\phi_i^{-1} = (\phi_{i,1}^{-1}, \cdots, \phi_{i,n+r}^{-1})$ with $x_{n+r} = \phi_{i,n+r}^{-1}(y_1, \cdots, y_{n+r}) = y_{n+r}$ and $x_j = \phi_{i,j}^{-1}(y_1, \cdots, y_{n+r}) = y_j - h_{i,j}(\phi_{i,j+1}^{-1}(y_1, \cdots, y_{n+r}), \cdots, \phi_{i,n+r}^{-1}(y_1, \cdots, y_{n+r}))$, for $j = n+r-1, \cdots, 1$.*

For instance, in the case of four variables, we have the inverse polynomial map ϕ_i^{-1} in the following abstract **general** form in term of variables,

$$
\begin{aligned}
&\phi_{i,4}^{-1}(y_1, \cdots, y_4) = y_4 \\
&\phi_{i,3}^{-1}(y_1, \cdots, y_4) = y_3 - h_{i,3}(y_4) \\
&\phi_{i,2}^{-1}(y_1, \cdots, y_4) = y_2 - h_{i,2}(y_3 - h_{i,3}(y_4), y_4) \\
&\phi_{i,1}^{-1}(y_1, \cdots, y_4) = y_1 - h_{i,1}(y_2 - h_{i,2}(y_3 - h_{i,3}(y_4), y_4), y_3 - h_{i,3}(y_4), y_4)
\end{aligned}
$$

In general, the total degree of $\phi_{i,j}^{-1}(y_1, \cdots, y_{n+r})$ increases very fast and the number of terms can be quite large as indicated by our later discussions. As shown in section 8, the number of terms in π^{-1} in our scheme is greater than 10^{92}. Therefore it is impractical to actually write down the polynomials $\phi_{i,j}^{-1}(y_1, \cdots, y_{n+r})$. However, if a point $(y_1', \cdots, y_{n+r}')$ is given, the value of the inverse map can be readily computed in the following **special** form in term of numbers.

Corollary 2 *Given a set of values* $(y'_1, \cdots, y'_{n+r}) \in \mathbf{K}^{n+r}$ *and a non-linear tame transformation* ϕ_i *as in the Definition of this section, then the values* $(x'_1, \cdots, x'_{n+r}) = (\phi^{-1}_{i,1}(y'_1, \cdots, y'_{n+r}), \cdots, \phi^{-1}_{i,n+r}(y'_1, \cdots, y'_{n+r})) \in \mathbf{K}^{n+r}$ *can be found by induction; first, we have* $x'_{n+r} = \phi^{-1}_{i,n+r}(y'_1, \cdots, y'_{n+r}) = y'_{n+r}$*, inductively we have* $x'_{j+1}, \cdots, x'_{n+r} \in \mathbf{K}$*, then we have* $x'_j = \phi^{-1}_{i,j}(y'_1, \cdots, y'_{n+r}) = y'_j - h_{i,j}(x'_{j+1}, \cdots, x'_{n+r})$ *for* $j = n + r - 1, \cdots, 1$*.*

Corollary 3 *Given the decomposition* $\pi = \prod_{i=1}^{i=n} \phi_i$ *where* ϕ_i *are tame automorphisms, then we have* $\pi^{-1} = \prod_{i=n}^{i=1} \phi_i^{-1}$*. Furthermore, if a set of values* $\{y'_j\}$ *is given, then we have* $\pi^{-1}(y'_1, \cdots, y'_{n+r}) = \prod_{i=n}^{i=1} \phi_i^{-1}(y'_1, \cdots, y'_{n+r})$*.*

3 Principle or Algorithm

Principle: Let q, n, r, s be positive integers. Let $n + r \geq 3$, and $\mathbf{K}$ a field of 2^q elements. Let the user select k tame transformations $\phi_k, \cdots, \phi_2, \phi_1$, such that the product $\pi = \phi_k \cdots \phi_2 \phi_1 = (\pi_1, \cdots, \pi_{n+r})$ defined a mapping $\mathbf{K}^n \mapsto \mathbf{K}^{n+r}$. Let $\hat{\pi} = (\pi_1(x_1, \cdots, x_n, 0, \cdots, 0), \cdots, \pi_{n+r}(x_1, \cdots, x_n, 0 \cdots, 0))$, and $f_i(x_1, \cdots, x_n) = \pi_i(x_1, \cdots, x_n, 0, \cdots, 0)$ for $i = 1, \cdots, n+r$.

The user will announce the map $\hat{\pi} = (f_1, \cdots, f_{n+r})$: $\mathbf{K}^n \mapsto \mathbf{K}^{n+r}$ and the field $\mathbf{K}$ of 2^q elements as the public key.

Given a plaintext $(x'_1, \cdots, x'_n) \in \mathbf{K}^n$. The sender evaluates $y'_i = f_i(x'_1, \cdots, x'_n)$. Then the ciphertext will be $(y'_1, \cdots, y'_{n+r}) \in \mathbf{K}^{n+r}$.

The legitimate receiver (i.e., the user) recovers the plaintext by $(x'_1, \cdots, x'_n, 0, \cdots, 0) = \phi_1^{-1} \cdots \phi_k^{-1}(y'_1, \cdots, y'_{n+r})$ (see **Corollaries** 2 & 3). The private key is the set of maps $\{\phi_1, \cdots, \phi_k\}$.

Remark: We may select ϕ_1 to be a linear embedding $\mathbf{K}^n \mapsto \mathbf{K}^{n+r}$ and ϕ_i elements in $\text{Tam}(\mathbf{K}^{n+r})$ for $i > 1$.

4 Component

We will give a report of an implementation (for other implementations, see http:// www.usdsi.com/ttm.html). We use a Component $\mathbf{Q}_8$ (see below). This component does **not** need to vary according to users. It can be made as part of the hardware. The user will select some other functions (see section 5) to make individual scheme non-traceable. The component in this section is example by nature, it is selected due to the theoretical clearness. Similar ones can be constructed.

The following definition will be used in the discussion of Component $\mathbf{Q}_8$,

Definition Let $q_1, \cdots, q_s$ be polynomials in variables $x_1, \cdots, x_t$. Let $\ell(x_1, \cdots, x_t)$ be a polynomial. If

$$Q(q_1(x_1, \cdots, x_t) \cdots, q_s(x_1, \cdots, x_t)) = \ell(x_1, \ldots, x_t)$$

Then Q is called a *generating polynomial* of ℓ (over $q_1, \cdots, q_s$) . Furthermore, if Q is of the minimal degree among all possible generating polynomials of ℓ, then it is called a *minimal generating polynomial* of ℓ, and its degree is called the *generating degree* of ℓ, in symbol *gendeg*(ℓ). If there is no such polynomial Q, then we define *gendeg*(ℓ)$= \infty$.

Now, let us define the Component $\mathbf{Q}_8$ as follows,

$\mathbf{Q}_8$: Let the field $\mathbf{K}$ be of 2^8 elements, and $a_i \neq 0$ for $i = 1, 2, 3$. Let

$q_1(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_4x_2 + a_1x_5$; $q_2(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_3x_4 + a_1x_6$;
$q_3(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_2x_5 + a_1x_7$; $q_4(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_4x_7 + a_1x_8$;
$q_5(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_1x_5 + a_1x_9$; $q_6(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_1x_2 + a_2x_{10}$;
$q_7(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_9x_2 + a_2x_{11}$; $q_8(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_9x_3 + a_1x_1$;
$q_9(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_1x_3$; $q_{10}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_1x_7 + a_1x_9$;
$q_{11}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_9x_4 + a_1x_1$; $q_{12}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_9x_7 + a_1x_1$;
$q_{13}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_3x_{11} + a_1x_{10}$; $q_{14}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{10}x_5 + a_1x_{11}$;
$q_{15}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{10}x_3$; $q_{16}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{10}x_2$;
$q_{17}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_7x_8 + a_1x_7$; $q_{18}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_7x_5 + a_1x_2$;
$q_{19}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_2x_3 + a_1x_7$; $q_{20}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_5x_8 + a_1x_5$;
$q_{21}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_5x_4 + a_1x_6$; $q_{22}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_3x_8$;
$q_{23}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_3x_5 + a_1x_8$; $q_{24}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_3x_7$;
$q_{25}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_6x_8 + a_3x_5$; $q_{26}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_6x_2$;
$q_{27}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_6x_5$; $q_{28}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_6x_7 + a_3x_2$;
$q_{29}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_2x_{11}$; $q_{30}(a_1, a_2, a_3, x_1, \cdots, x_{12}) = x_{11}x_4 + a_1x_{10}$;
$q_{31}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{10}x_7 + a_1x_{11}$; $q_{32}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = (x_3 + x_5)x_6 + a_1x_4$;
$q_{33}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{11}x_8$; $q_{34}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{10}x_8$;
$q_{35}(a_1, a_2, a_3, x_1, \cdots, x_{11}) = x_{11}x_7 + a_1x_{10}$;

Then the following Q_8 is a minimal generating polynomial of $a_1^{14}(x_1x_{11} + x_{10}x_9)$ with degree 8 in q_i,

$$\begin{aligned} Q_8 = \; & (q_5q_{13} + q_8q_{14})(q_{19}q_{32} + q_2(q_{18} + q_{24}))^2(q_{20}q_{19} + q_{23}q_{18})) + (q_{32}q_3 + (q_{18} + q_{24})q_{21})^2 \\ & (q_{22}q_{19} + q_{23}q_{24})(q_9q_{13} + q_8q_{15}) + a_1^8((q_{25}q_{26} + q_{27}q_{28})(q_6q_{29} + q_7q_{16}) \\ & +(q_{10}q_{30} + q_{11}q_{31})(q_{17}q_1 + q_{18}q_4)) + a_1^8a_2^4(q_6q_{33} + q_{34}q_7 + q_5q_{35} + q_{14}q_{12}) \\ & = a_1^{14}(x_1x_{11} + x_{10}x_9) \end{aligned}$$

■

For the convenience of discussion in the next section, let us define

Definition An invertible linear transformation $\phi_1 = (\phi_{1,1}, \cdots, \phi_{1,n+r})$ is said to be of type A if

$$\phi_{1,i} = \sum_{j=1}^{n+r} a_{i,j}x_j + b_i$$

then

(1)For $i = 1, \cdots, n$, we always have $b_i \neq 0, a_{i,j} = 0$, for $j = n+1, \cdots, n+r$ and at least half of the remainning $a_{i,j}$ are non-zero.

(2)For $i = n+1, \cdots, n+r$, we always have $\phi_{1,i} = x_i$.

Implementation

We will give an implementation of TTM cryptosystem based on the above example. In this implementation, we will assume that $a_i = 1$ for simplicity.

We shall use the notations of above $\mathbf{Q}_8$, q_i. Let $K = GF(2^8)$, $n \geq 30$ and $m = n + 52$. We have four maps, $\phi_1, \phi_2, \phi_3, \phi_4$ with the map $\pi = \phi_4\phi_3\phi_2\phi_1$ where ϕ_1 is an affine linear map of K^n to the subspace of K^m with the last 52 coordinates zeroes, ϕ_2, ϕ_3 tame maps and ϕ_4 an affine linear map of K^m to K^m.

The rationalities of designing this encryption system are as follows. We may call the compositions of two tame transformations $\phi_3\phi_2$ the kernel of the system. It will play the role of *trap-door* of the system and it can be *individualized.* The two other transformations on two sides are linear and hide the whole system from an attacker.

For readers convenient, we define polynomials P_1, P_2, P_3 as for $j = 1, 2, 3$.

$$\begin{aligned} P_j &= P_j(y_{m-58}, \cdots, y_{m-55}, y_{m+1-8j}, \cdots, y_{m+8-8j}, y_{m-46}, \cdots, y_{m-24}) \\ &= Q_8(y_{m-58}, \cdots, y_{m-55}, y_{m+1-8j}, \cdots, y_{m+8-8j}, y_{m-46}, \cdots, y_{m-24}) \end{aligned}$$

and P_4 as $P_4 = P_4(y_{m-58}, \cdots, y_{m-24}) = Q_8(y_{m-58}, \cdots, y_{m-24})$. Then we select suitable $\beta_{ij} \neq 0$ for $i, j = 1, \cdots, 4$ such that $R_i = \sum_j \beta_{ij} P_j$ are linearly independent.

We should look at the composition $\phi_3\phi_2$ which can be expressed as

$$\phi_3\phi_2 = \begin{cases} y_1 = x_1 + R_1 \\ \quad = x_1 + \beta_{14}(x_{m-62}x_{m-52} + x_{m-53}x_{m-54}) + \sum_{j=1}^{j=3} \beta_{1j}(x_{m-62-2j}x_{m-52} + x_{m-61-2j}x_{53}) \\ y_2 = x_2 + f_2(x_1) + R_2 \\ \quad = x_2 + f_2(x_1) + \beta_{24}(x_{m-62}x_{m-52} + x_{m-53}x_{m-54}) \\ \qquad + \sum_{j=1}^{j=3} \beta_{2j}(x_{m-62-2j}x_{m-52} + x_{m-61-2j}x_{53}) \\ y_3 = x_3 + f_3(x_1, x_2) + R_3 \\ \quad = x_3 + f_3(x_1, x_2) + \beta_{34}(x_{m-62}x_{m-52} + x_{m-53}x_{m-54}) \\ \qquad + \sum_{j=1}^{j=3} \beta_{3j}(x_{m-62-2j}x_{m-52} + x_{m-61-2j}x_{53}) \\ y_4 = x_4 + f_4(x_1, x_2, x_3) + R_4 \\ \quad = x_4 + f_4(x_1, x_2, x_3) + \beta_{44}(x_{m-62}x_{m-52} + x_{m-53}x_{m-54}) \\ \qquad + \sum_{j=1}^{j=3} \beta_{4j}(x_{m-62-2j}x_{m-52} + x_{m-61-2j}x_{53}) \\ y_5 = x_5 + f_5(x_1, \cdots, x_4) \\ \cdots \\ y_{m-59} = x_{m-59} + f_{m-59}(x_1, \cdots, x_{m-60}) \\ y_{m-58} = q_1(x_{m-62}, \cdots, x_{m-52}) = x_{m-58} + x_{m-59}x_{m-61} \\ \cdots \\ y_{m-52} = q_7(x_{m-62}, \cdots, x_{m-52}) = x_{m-52} + x_{m-54}x_{m-61} \\ y_{m-51} = q_8(x_{m-62}, \cdots, x_{m-52}) \\ \cdots \end{cases}$$

$$\begin{cases} y_{m-24} = q_{35}(x_{m-62}, \cdots, x_{m-52}) \\ y_{m-23} = q_5(x_{m-64}, x_{m-61}, \cdots, x_{m-55}, x_{m-63}, x_{m-53}, x_{m-52}) \\ \cdots \\ y_{m-16} = q_{12}(x_{m-64}, x_{m-61}, \cdots, x_{m-55}, x_{m-63}, x_{m-53}, x_{m-52}) \\ y_{m-15} = q_5(x_{m-66}, x_{m-61}, \cdots, x_{m-55}, x_{m-65}, x_{m-53}, x_{m-52}) \\ \cdots \\ y_{m-8} = q_{12}(x_{m-66}, x_{m-61}, \cdots, x_{m-55}, x_{m-65}, x_{m-53}, x_{m-52}) \\ y_{m-7} = q_5(x_{m-68}, x_{m-61}, \cdots, x_{m-55}, x_{m-67}, x_{m-53}, x_{m-52}) \\ \cdots \\ y_m = q_{12}(x_{m-68}, x_{m-61}, \cdots, x_{m-55}, x_{m-67}, x_{m-53}, x_{m-52}) \end{cases}$$

where f_i's are random quadratic polynomials such that the vector space dimension of all homogeneous degree 2 parts of the above system is m.

As usual we further require that $\pi(0, \cdots, 0) = (0, \cdots, 0)$. Then π is the public key, while $\{\phi_1, \phi_2, \phi_3, \phi_4\} is the private key.$ ■

The user selects ϕ_4 to be an invertible linear transformation satisfying condition $**$ in the following way, where $\pi = (\pi_1, \cdots, \pi_m)$,

$$** : \pi = \phi_4\phi_3\phi_2\phi_1, \text{ and } \pi_i(0, \cdots, 0) = 0$$

The field $\mathbf{K}$ and polynomials $f_i(x_1, \cdots, x_n) = \pi_i(x_1, \cdots, x_n, 0, \cdots, 0)$ for $i = 1, \cdots, m$ will be announced as the public key. The private key is the set of maps $\{\phi_1, \phi_2, \phi_3, \phi_4\}$.

Let $(x'_1, \cdots, x'_n) \in \mathbf{K}^n$ be the plaintext. The sender evaluates $y'_i = f_i(x'_1, \cdots, x'_n)$. Then the resulting $(y'_1, \cdots, y'_m) \in \mathbf{K}^m$ will be the ciphertext.

The legitimate receiver (i.e., the user) recovers the plaintext by $(x'_1, \cdots, x'_n, 0, \cdots, 0) = \phi_1^{-1} \cdots \phi_4^{-1}(y'_1, \cdots, y'_m)$ which can be done easily according to **Corollaries** 2 &3. ■

5 Plaintexts, Users and Compactness

In this section let us assume that $n = 48, m = 100$ for numerical computations.

Let us count the possible **number of plaintexts**. Since the number of plaintexts is the number of choices for $x'_1, \cdots, x'_{48}$, we see that there are 2^{382} such plaintexts.

Of equal importance to having a large number of possible plaintexts is having lots of possible **users**. In order to allow for many such users, we first get an expression for this number in terms of m and 48. This amounts to counting the number of automorphisms π of the form $\pi = \phi_1\phi_2\phi_3\phi_4$. Assuming that a negligible proportion of these automorphisms π have more than one representation $\pi = \phi_1\phi_2\phi_3\phi_4 = \phi'_1\phi'_2\phi'_3\phi'_4$, the number of users is asymptotic to (choices for ϕ_4) $\times$ (choices for ϕ_3) $\times$ (choices for ϕ_2) $\times$ (choices for ϕ_1). The condition of type A for ϕ_1 is not a big restriction. We may use the possible numbers for the invertible linear transformations as an estimation for ϕ_1, ϕ_4. The number of invertible linear transformations ϕ_1 is $\prod_{j=0}^{n-1}(2^{8n} - 2^{8j}) = 2^{8n(n-1)/2}\prod_{j=1}^{n}(2^{8j} - 1)$. For our selection of $n = 48, n+r = 100$, it is easy to see that the number for $\phi_1 > 2^{9024}$. A similar count of terms of ϕ_4 results in the total possible number of users $> 2^{61424}$.

Now let us look at the **compactness** of the scheme. The end results are 100 quadratic polynomials $(f_1, \cdots, f_{100})$ in 48 variables $x_1, \cdots, x_{48}$. It is easy to see that the number of terms of polynomials of degree 2 is (51)(48)/2! and we have 100 polynomials, therefore the total number of terms is $122,400$. This is the size of the public key. The expense to the sender is mainly in evaluating polynomials $y'_i = f_i(x'_1, \cdots, x'_{48})$. Note that our scheme can also be computed in a parallel way. Thus the process can be sped up. On the receiver's side, the total number of terms for $\phi_1, \phi_2, \phi_3, \phi_4$ is $17,000$ (the corresponding size of the private key for TTM 3.2 is 184 bytes). The legitimate receiver needs to find $(x'_1, \cdots, x'_{48})$ from $\phi_4\phi_3\phi_2\phi_1(x'_1, \cdots, x'_{48})$ (according to **Corollaries** 2 & 3) which is not expensive.

6 Technique Report

Following the principle of this article, there are several software implementations. For the convenience of discussions, the method will be called "tame transformation method" (TTM). There are versions TTM 1.9, TTM 2.1, TTM 2.3, TTM 2.5 TTM 2.5, TTM 3.0, TTM 3.2, TTM 4.3 and TTM 4.5 available. They use C Language. For GF(2^8), the rates of expansion of data are 1.4, 1.56, 1.63, 1.5, 2.66, 3.5, 2.2 and 3.2 respectively. They have been used on various machines listed below,

266 *Mhz PowerPC* 750.
200 *Mhz PowerPC* 604*e w*/1024*K cache.*
225 *Mhz PowerPC* 603*e w*/256*K cache*,
167 *Mhz Ultrasparc w*/512*K cache.*
167 *Mhz Pentium w*/512*K cache.*
400 *Mhz PowerPC G*4

The software TTM 2.1, 2.3, and 2.5 are on 200 Mhz PowerPC 604e (w/1024K cache: virtual memory off), the softwares TTM 3.0, TTM 3.2 are on 266 Mhz PowerPC 750 and the softwares TTM 4.3, TTM 4.5 are on 400 Mhz PowerPC G4. Their speeds are listed in the following;

The implementation speed depends on the speed to compute a*b+c where a,b and c are in the 8-bits finite field. For TTM 3.2, every 66.5 repetitions of this computation will process one bit of information, while each computation requires 3.75 cycles.

The decrypting speed is in general faster than the encrypting speed (the decrypting speed for TTM 3.2 is 8,271,298 bit/sec). For the user who knows the private key, the speed of encoding can be increased to 8,271,298 bit/sec. The PC software TTM 3.2 is faster than a possible hardware implementation for RSA 1024. While the speed of TTM 4.5 is about 1/5 the speed of the very fast secret key system AES.

It is possible to encrypt voice communications(64,000 bit/sec) by those softwares on an ordinary PC. Note that in comparison, RSA toolkit BSAFE 3.0 for 1024 is 7 K bit/sec.

Smart Card Applications: for an "untrustworthy" PC, the tentative report of our Lab is that the requirement of RAM is 1k bytes for the program and private key with a speed exceeding 50 k bit/sec. For a "trustworthy" PC, one may simply put the set up numbers

	speeds of software implementations
TTM 1.9,	94,939b/s
TTM 2.1,	106,224b/s.
TTM 2.3,	207,000b/s
TTM 2.5,	300,000b/s.
TTM 2.8,	658,323 bit/sec
TTM 3.0,	1,001,774 bit/sec
TTM 3.2,	1,626,944 bit/sec
TTM 4.3,	18,000,000 bit/sec
TTM 4.5,	33,323,672 bit/sec

Table 1:

in the card for the machine to generate the private key. Then it requires only 30–40 bytes memory in the smart card and less than 1 second to set up

7 Useful Properties of the Scheme

Error-Detect Function

Upon receiving the ciphertext $(y_1', \cdots, y_m')$, the user applies **Corollaries** 2 & 3 to decode it and get $(\bar{x}_1, \cdots, \bar{x}_m)$. If one of $\bar{x}_{n+1}, \cdots, \bar{x}_m$ is not zero, then there must be an error.

Master Key Function

Let a group of indices, S , be a few extra indices $m+1, m+2, \cdots$ added. Select ϕ_4 such that the corresponding subspace generated by x_i with $i \in S$ and the subspace generated by x_j with $j \notin S$ are both invariant. The original public key scheme gives a master key. A subordinate key can be produced by deleting all f_i's with $i \in S$.

A different way to produce a master key is to use the polynomials $q_1, \cdots, q_{35}$ of section 4, and extended it to $q_{36}, \cdots, q_{35+s}$ and to find a polynomial $Q'(q_1, \cdots, q_{35}, \cdots, q_{35+s})$, such that both Q' and its specialization $Q'(q_1, \cdots, q_{35}, 0, \cdots, 0)$ can be used to construct a public key scheme. We require that ϕ_1 to keep space $\{(c_1, \cdots, c_m, 0, \cdots, 0) : c_i \in \mathbf{K}\} = \mathbf{K}^m \times 0 \times \cdots \times 0$ invariant and use the specialization $x_i \mapsto 0$ for $i = m+1, \cdots, m+s$ to create a subordinate key from the original key (i.e., the master key).

The 'master key-subordinate key' relation can be broken by alternating any one of the linear transformations ϕ_1, ϕ_4 involved.

Signatures

The map $\hat{\pi}$ is not an onto map. However, we may restrict the map to a suitable linear subspace V. Let us use the example of this article to illustrate the method. Let V be a coordinate space which is $\{(d_1, \cdots, d_n) : d_i \in \mathbf{K}$ and $d_i = 0$ if $i = n-5, n-6, n-8, n, n-19, n-20, n-22, n-14\} \subset \mathbf{K}^n$ with $j = \dim(V)$. Consider V as a subspace of $\mathbf{K}^m$ by adding suitable many zeroes for the last $m-n$ coordinates. Let $\bar{V} = \phi_1^{-1}(V)$. We will require that ϕ_4 induces a linear transformation on V. Let $\tau : \mathbf{K}^m \mapsto V$ be the projection. Clearly $\tau\hat{\pi}$ is a one to one and onto map from $\bar{V}$ to the j-dimensional affine space. Moreover, the map is *tame*, and its inverse at a point $(y'_1, \cdots, y'_j)$ can be found. The inverse at $(y'_1, \cdots, y'_j)$ forms a *signature*.

8 Cryptanalysis for the Scheme

For the first four analysis, we take $n = 64, m = 100$.

I. Direct Methods

There is no known way to recover the private key $\{\phi_4, \cdots, \phi_1\}$ from the public key $\hat{\pi}$ and the field $\mathbf{K}$. There are three other direct ways to attack the scheme: (1) use 'inverse formula' for power series to find polynomial expressions of π^{-1} ([10]). Note that only $\hat{\pi}$ is given, since $\hat{\pi}^{-1}$ does not exist theoretically, there is no way to find it, (2) let x_i be a polynomial, g_i, of $\{y_j\}$ with indeterminate coefficients for all i. Do enough experiments using $\{x_i\}$ to determine $\{y_j\}$ and then solve the system of linear equations in indeterminate coefficients to find polynomials g_i, or (3) using 'resultant' to the expression $y'_i = f_i(x'_1, \cdots, x'_{64})$ to eliminate all x'_i except one, say x'_j, and recover the expressions of x'_j in terms of $y'_1, \cdots, y'_{100}$.

For the method (1), note that the form of the map π is not given to the public, the attacker has to guess π correctly. Furthermore, it follows from **Corollary 2** and the explicit expression of ϕ_2 in the scheme that we have,

$$max\{deg_{y_1}\ \phi_{2,j}^{-1}(y_1, \cdots, y_{100})\} \geq 2^8$$

Since ϕ_1, ϕ_4 are linear transformations, the **theoretic total number of terms** in π^{-1} is $100(\prod_{i=1}^{100}(2^8 + i))/100! > 10^{92}$. It is too large to be practical. To use the method (2), the attacker has to give an estimation of the degrees of g_i. According to our previous analysis, there are too many terms in g_i's for the method to be useful. As for the method (3), the resultant is only practical for polynomials of very few variables, it is impractical in our scheme.

At this moment, the above three direct methods are ineffective. The only possible way of attacking is to recover ϕ_i's or their equivalent forms.

II. Search for Polynomial Relations

Although polynomials $\{f_1, \cdots, f_{100}\}$ are linearly independent, the attacker may search for polynomial relations, which are linear relations of monomials, among them. Knowing the recipe of the construction of the public key scheme, the attacker may launch a '**step by step search**' to search for useful polynomials in the ring $\mathbf{K}[f_1, \cdots, f_{100}]$. One can show that the attacker has to consider the vector space of all polynomials of degree ≤ 8 in $f_1, \cdots, f_{100}$. The number of those polynomials of degree ≤ 8 is $C_8^{108} \approx 3.52(10^{11})$ which is in a vector space of dimension $2.69(10^{16})$. The dimension is too high to be handled by present technology. To store these data, if we may mobilize the world population, then every human being, young or old, will use 2 billion gigabytes memory. In comparison, the San Diego Supercomputer Center hosts the largest data-storage system of 86,000 gigabytes. The task is huge. We can select polynomials Q_8 with higher degrees to defend the scheme if necessary.

III. Identify the Highest Homogeneous Parts

The attacker may try to find $\phi_4^{-1}\hat{\pi}$ first. The very first step is to identify the highest homogeneous part. After some deeper analysis of the problem, it can be shown that just to do the very first step, it requires $82(10^{317})$ mips (one million instruction per second) years.

IV. Brute Force Attack

The attacker may use linearly independent linear polynomials $\{v_1, \cdots, v_{48}$ to express $\{f_1, \cdots, f_{100}\}$. Then the attacker assign random values from the field K for the subsets of $\{v_1, \cdots, v_{48}\}$ to see if the assigned values are correct. It is easy to see that the possibilities is $2^{8*48} = 2^{384} = 10^{128}$. **Assuming** it only take one clock cycle to test if a set of 48 random numbers is correct, the attacker still need 3×10^{114} mips years to crack the scheme. In comparison, it requires 3×10^{20} mips years to cracked RSA 2048.

V. The Method of XL

In the article [8], Nicolas Courtois, Adi Shamir, Jacques Patarin and Alexander Klimov propose a method named "XL" which gives an "*efficient algorithm for solving overdefined systems of multivariate polynomial equations*". In the abstract, they state "*we then develop an improved algorithm called XL which is both simpler and more powerful than relinearization*".

However, it is largely a misunderstanding of mathematics to make such a claim. An analysis based on Hilbert-Serre theory on "Hilbert functions" shows the inefficiency of the said method in [21].

VI. L. Goubin and N.T. Courtois' Attack

In the paper [13] published in "Asiacrypt 2000", L. Goubin and N.T. Courtois propose an attack on TTM cryptosystem.

They established a formula of complexity of their attack $q^{\lceil \frac{m}{n} \rceil r} \times m^3$ for their own system TPM which is different from TTM, where q is the number of elements in the ground field, m is the length of the ciphertext, n is the length of plaintext and r is the number of variables used in the quadratic forms of the first few polynomials. They mistakenly assume that $r = 2$, which is clearly false as can be computed to be $r = 4$ in the example of this article. (cf [22]). For the present example the complexity $= 2^{84}$ according to their attack, which means the present example is much higher than the complexity of a strong encryption system with a complexity of 2^{80}. The said attack is very ineffective contrast to what they claimed.

We may consider a "MinRank(r)" attack on TTM system, and show that it is ineffective [22].

9 Summary

The present implementation scheme can withstand all known attacks. By its nature, the algorithm is less cumbersome to use than methods that are number theory based. The drawbacks of this system are the huge size of the keys and the rate of data expansion. However, for parctical usage, both can be reduced to manageable numbers. Furthermore, it has the novel functions of error-detect and master key. We wish that this algorithm will provide a new direction of research.

APPENDIX

Acknowledgement

We wish to express our thanks to J. M. Acken, A. Odlyzko, H. Lenstra, P. Montgomery, E. Croot, C. Bajaj, S. Wagstaff, F. Chao, R. Osawa, P. Huang for discussions.

After we sent out our original draft in 1995, P. Montgomery responded in showing us a successive attack on the example of four variables in that draft. We produced another version which defended against the 'analysis of the highest homogeneous parts' to stand the attack proposed by P. Montgomery. Then A. Sathaye launched a 'step by step search', i.e., searching for relations among all monomials of $\{f_i\}$ with some fixed degree, which could theoretically crack our second version. Only then did we understand that the attack of P. Montgomery was the beginning of a 'step by step search'. The point is that the final polynomials $(f_1, \cdots, f_{n+r})$ are of various degrees, and they can be grouped and analyzed according to their degrees. In our previous versions, for a particular degree, there are only a few polynomials (or it is the same to say, a small dimensional vector space). Those polynomials can be discovered by a 'step by step search' even though they were covered up at the beginning by a linear transformation of the vector space generated by $\{f_i\}$. Therefore, the previous public key system would dissolve step by step theoretically.

Due to our intention of including a 'master key function' (see section 7) in the system, we had considered a *specialization* $x_{n+r} \mapsto x_i$. Independently, from the point of view of *embedding theory* (cf [1], [22]), A. Sathaye suggested that we should consider $x_{n+r} \mapsto 0$. These two approaches were identical up to a linear transformation. Very soon we solved the technical problem involved (see section 7). In our present public key scheme, the resulting polynomials are of degree two uniformly and their degree two homogeneous parts are linearly independent (see the end of section 4).

Applying 'step by step search' to the present public key scheme, an attacker will have to consider a set of $3.52(10^{11})$ elements in a vector spaces of dimensions $2.69(10^{16})$ see section 8, **II**). The dimension is too high to be handled by the present technology. Moreover, since the encoding scheme uses 'embedding maps' without inverses, it is impossible to crack this scheme by looking for inverses. The present scheme can withstand known attacks. We are especially grateful to A. Sathaye for the enlightening discussions and for checking our computations in section 5.

References

[1] Abhyankar, S.S. and Moh, T. *Embeddings of the line in the plane.* Journal für die reine und angewandte Mathematik., pp 148-166, vol 276, 1975.

[2] Bajaj, C. Garrity, T. Warren, J. *On the Application of Multi-Equational Resultants.* Purdue University, Dept of C. S. Technical Report CSD-TR-826, 1988.

[3] Bass, H. Connell, E.H. Wright, D.L. *The Jacobian conjecture: reduction of degree and formal expansion of the inverse.* Bull. Amer. Math. Soc. pp 287-330 no. 2 (N.S.) 7, 1983.

[4] Berlekamp, E.R. *Factoring polynomials over finite fields.* Bell System Tech. J. pp 1853-1859, vol 46, 1967.

[5] Brandstrom, H. *A public-key cryptosystem based upon equations over a finite field.* Cryptologia, pp 347-358, vol 7, 1983.

[6] Brent, R., and Kung, H. *Fast Algorithms for Manipulating Formal Power Series.* Journal of the ACM, pp 581-595, vol 25 no 4, 1978.

[7] Cohen, Henri *A Course in Computational Algebraic Number Theory.* Springer-Verlag. Berlin, 1993.

[8] Courtois, N. Shamir, A. Patarin, J. and Klimov, A. *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations.* Eurocrypt'2000

[9] Canny, John F. *Complexity of Robot Motion Planning.* The MIT Press, Cambridge, Massachusetts, 1988.

[10] Dickerson, Mathew *The inverse of an Automorphism in Polynomial Time.* J. Symbolic Computation, vol 13. 209-220, 1992.

[11] van der Gathen, J. *Functional decomposition of polynomials: the tame case.* J. Symbolic Comp. 9, 281-299, 1990.

[12] van der Gathen, J. *Functional decomposition of polynomials: the wild case.* J. Symbolic Comp. 10, 437-452, 1990.

[13] Goubin, L. and Courtois, N.T. *Cryptanalysis of the TTM Cryptosystem.* Asiacrypt 2000, Dec 2000.

[14] Imai, H. Matsumoto *Algebraic methods for constructing asymmetric cryptosystems, Algebraic and Error-Correcting Codes.* Prod. Third Intern. Conf., Grenoble, France, Springer-Verlag, 108-119, 1985.

[15] Lidl, R *Finite fields.* Addison-Wesley, Reading, Massachusetts, 1983.

[16] Lidl, R *On Cryptosystems Based on Polynomials and Finite Fields.* Advances in Cryptology (Proceedings of EUROCRYPT 84), pp 10-15, 1983.

[17] Lucier, B. *Cryptography, Finite Fields, and AltiVec.* http://www.altivec.org/articles/

[18] Moh, T. *On the Classification Problem of Embedded Lines in Characteristic p.* Algebraic Geometry and Commutative Algebra *in honor of M. Nagata*, vol I, pp 267-280, Kinokuniya, Kyoto, Japan, 1988.

[19] Moh, T. *A Public Key System with Signature and Master Key Functions.* Communications in Algebra, 27(5), 2207-2222 (1999).

[20] Moh, T. *A Fast Public Key System With Signature And Master Key Functions.* CrypTEC'99 (Proc. International Workshop on Cryptographic Techniques & E-commerce), City University of Hong Kong Press, July, 1999.

[21] Moh, T. *On the Method of XL and Its Inefficiency Against TTM.* http://www.usdsi.com/ttm

[22] MOH, T. *On the the Goubin-Courtois Attack on TTM.* "Cryptology ePrint Archive", a copy can also be obtained at http://www.usdsi.com/ttm

[23] NAGATA, M *On the automorphism group of* $\mathbf{K}[X, Y]$., vol 5, Kinokuniya, Tokyo, Japan, 1972.

[24] NIEDERREITER, H. *New Deterministic Factorization Algorithms for Polynomials over Finite Fields.* Contemporary Mathematics (Finite Fields) (AMS), vol 168, 1993.

[25] PATARIN, J *Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms.* Advances in Cryptology- Euricrypt '96, Springer-Verlag, 33-48, 1996.

[26] RIVEST, R.L. SHAMIR, A. AND ADLEMAN, L.M. *A Method for Obtaining Digital Signatures and Public Key Cryptosystems.* Communications of the ACM 21(2), 120-126, Feb 1978.

[27] VAN DER KULK, W. *On polynomial rings in two variables,* Nieuw Archief voor Wiskunde. vol 3, I(1953).

[28] PATARIN, J. *Cryptanalysis of Matsumoto and Imai Public Key Scheme.* Eurocrypt 88, Crypto 95, Springer Verlag, pp. 248-261