PUBLIC QUADRATIC POLYNOMIAL-TUPLES FOR EFFICIENT SIGNATURE-VERIFICATION AND MESSAGE-ENCRYPTION

Tsutomu Matsumoto

Hideki Imai

Division of Electrical and Computer Engineering
YOKOHAMA NATIONAL UNIVERSITY
156 Tokiwadai, Hodogaya, Yokohama, 240 Japan

Abstract This paper discusses an asymmetric cryptosystem C^* which consists of public transformations of complexity $O(m^2n^3)$ and secret transformations of complexity $O((mn)^2(m+\log n))$, where each complexity is measured in the total number of bit-operations for processing an mn-bit message block. Each public key of C^* is an n-tuple of quadratic n-variate polynomials over $GF(2^m)$ and can be used for both verifying signatures and encrypting plaintexts. This paper also shows that for C^* it is practically infeasible to extract the n-tuple of n-variate polynomials representing the inverse of the corresponding public key.

I. INTRODUCTION

With the aid of public-key cryptography[1], how much computation is sufficient to keep the authenticity and the confidentiality of digital data? Reducing the computational complexity implies wider and deeper utilization of the fascinating nature of public-key cryptography. This paper gives an answer to this challenging question by constructing an asymmetric cryptosystem C^* (called c-star) which consists of public transformations of complexity $O(m^2n^3)$ and secret transformations of complexity $O((mn)^2(m + \log n))$, where each complexity is measured in the total number of bit-operations for processing a message block of mn bits.

Each public key of C^* is an n-tuple

$$F = [F^{(0)}(x_0, \ldots, x_{n-1}), \ldots, F^{(n-1)}(x_0, \ldots, x_{n-1})]$$

C.G. Guenther (Ed.): Advances in Cryptology - EUROCRYPT '88, LNCS 330, pp. 419-453, 1988.

of quadratic n-variate polynomials over $K = GF(2^m)$, and the corresponding public transformation translates a message block $\xi \in K^n$ into another message block $\eta \in K^n$, by evaluating F at ξ. Here the term "quadratic polynomials " means "polynomials of degree 2", and the degree $deg(P)$ of a polynomial

$$P(x_0, \ldots, x_{s-1}) = \sum \{P_{i_0 \cdots i_{s-1}} x_0^{i_0} \cdots x_{s-1}^{i_{s-1}} | i_0, \cdots, i_{s-1} \geq 0\},$$

is determined by

$$\deg(P) = \begin{cases} \max\{i_0 + \cdots + i_{s-1} | P_{i_0 \cdots i_{s-1}} \neq 0\}; & P \neq 0, \\ -\infty; & P = 0. \end{cases}$$

Since F is defined to be representing a bijection, F can be used for both verifying signatures and encrypting plaintexts.

Small Example 1.

$m = 1, n = 8$

$$\begin{cases} y_0 = F^{(0)}(x_0, \ldots, x_7) = x_0 + x_1 + x_3 + x_7 + x_0x_1 + x_0x_2 + x_0x_4 + x_0x_5 + x_0x_6 + x_0x_7 + x_1x_4 + x_1x_6 \\ \qquad + x_1x_7 + x_2x_6 + x_3x_4 + x_3x_5 + x_3x_7 + x_4x_5 + x_5x_6 + x_5x_7 \\ y_1 = F^{(1)}(x_0, \ldots, x_7) = x_1 + x_2 + x_4 + x_6 + x_0x_3 + x_0x_6 + x_0x_7 + x_1x_3 + x_1x_4 + x_1x_6 + x_2x_5 + x_2x_7 \\ \qquad + x_3x_4 + x_3x_7 + x_4x_6 + x_4x_7 + x_6x_7 \\ y_2 = F^{(2)}(x_0, \ldots, x_7) = 1 + x_0 + x_1 + x_2 + x_3 + x_5 + x_6 + x_0x_1 + x_0x_2 + x_0x_5 + x_1x_2 + x_1x_4 + x_1x_6 \\ \qquad + x_1x_7 + x_2x_6 + x_2x_7 + x_3x_5 + x_3x_6 + x_3x_7 + x_4x_5 + x_5x_6 + x_5x_7 \\ y_3 = F^{(3)}(x_0, \ldots, x_7) = x_0 + x_2 + x_3 + x_7 + x_0x_3 + x_0x_5 + x_1x_4 + x_1x_5 + x_1x_6 + x_1x_7 + x_2x_3 + x_2x_4 \\ \qquad + x_2x_7 + x_3x_4 + x_3x_7 \\ y_4 = F^{(4)}(x_0, \ldots, x_7) = 1 + x_0 + x_1 + x_2 + x_6 + x_7 + x_0x_2 + x_0x_4 + x_0x_5 + x_1x_3 + x_1x_7 + x_2x_6 + x_3x_4 \\ \qquad + x_3x_5 + x_3x_6 + x_4x_5 + x_4x_6 + x_4x_7 + x_5x_6 + x_5x_7 + x_6x_7 \\ y_5 = F^{(5)}(x_0, \ldots, x_7) = x_4 + x_6 + x_0x_1 + x_0x_2 + x_0x_3 + x_1x_2 + x_1x_3 + x_1x_5 + x_2x_6 + x_3x_4 + x_3x_7 \\ y_6 = F^{(6)}(x_0, \ldots, x_7) = 1 + x_0 + x_2 + x_3 + x_7 + x_0x_1 + x_0x_4 + x_1x_3 + x_1x_4 + x_1x_6 + x_2x_3 + x_2x_4 \\ \qquad + x_2x_6 + x_3x_4 + x_3x_5 + x_3x_7 + x_4x_5 + x_4x_6 + x_4x_7 + x_5x_6 + x_5x_7 + x_6x_7 \\ y_7 = F^{(7)}(x_0, \ldots, x_7) = x_0 + x_1 + x_4 + x_5 + x_7 + x_0x_1 + x_1x_3 + x_1x_5 + x_2x_6 + x_2x_7 + x_3x_5 + x_3x_6 \\ \qquad + x_3x_7 + x_4x_5 + x_5x_6 + x_5x_7 \end{cases}$$

Solving a randomly selected system of multivariate polynomial equations is widely recognized as a very difficult problem[2], and indeed it is NP-hard even in the case[3] of quadratic equations over $GF(2)$. Therefore, the idea of using multivariate polynomials as public keys has attracted several cryptographers. The problem they must answer is how to construct a tuple of multivariate polynomials which (1) represents a bijection and (2) is easy to evaluate but difficult to invert.

For example, Matsumoto *et al.*[4] have proposed the idea of starting from univariate polynomials over large finite fields. And they have developed this idea into a general construction method called *the algorithm composition method* [5], on which the cryptosystem C^* is also based.

On the other hand, Fell and Diffie[6] have proposed an approach of combining DES-like structure into multivariate polynomials and concluded that their approach seems not to produce polynomial-tuples satisfying the request (2) since the degrees of the original and the inverse polynomial-tuple are the same. Here, the degree $deg(Q)$ of a polynomial-tuple $Q = [Q^{(0)}, \ldots, Q^{(t-1)}]$ is defined as $max\{deg(Q^{(j)}) | j = 0, \cdots, t-1\}$.

Also, Zhou[7,8] have proposed a cryptosystem using polynomial-tuples over $GF(2)$ constructed by a method similar to Fell's and Diffie's.

However, at least by the method due to Matsumoto *et al.* it is possible to systematically construct low-degree multivariate polynomial-tuples whose inverse polynomial-tuples have very high degree. Actually, this paper shows that for C^* it is practically infeasible to extract the n-tuple G of n-variate polynomials representing the inverse of the corresponding public key F.

In the following, Chapter II describes the definition of the asymmetric cryptosystem C^* and three important theorems for it. Chapter III develops concrete algorithms for implementing C^* and proves Theorem 2, which states the operational complexity of C^* . Chapter IV describes the process of deriving C^* and proves both Theorem 1, which states the consistency of the definition of C^* , and Theorem 3, which guarantees a certain security aspect of C^* . And Chapter V concludes the paper.

II. THE PROPOSED ASYMMETRIC CRYPTOSYSTEM

Definition 1. The asymmetric cryptosystem C^* is defined by the following public items P1,...,P5 and secret items S1,...,S4.

[Public Items]

P1. A positive integer m and an integer $n \geq 3$, but $n \neq 4$;

P2. A finite field K of order $q = 2^m$ with an adder and a multiplier;

P3. The set of message blocks K^n, which is the n-dimensional vector space consisting of all n-tuples over K;

P4. Each public key is an n-tuple F of quadratic n-variate polynomials over K;

P5. The public transformation algorithm PA, which transforms a message block $\xi \in K^n$ into another message block $\eta = PA(F, \xi) \in K^n$ by evaluating F at ξ.

[Secret Items]

S1. A ν-degree extension field $L_{(\nu)}$ of K and a K-isomorphism $\psi_{(\nu)}$ from K^ν to $L_{(\nu)}$ for each integer $\nu = (2\lambda + 1)2^\rho$ with $\lambda \geq 1$ and $\rho \geq 0$;

S2. Each secret key is a tuple $\Gamma = [S^R, T^R, \pi, \Theta]$:

S2-1. Two n-tuples S^R and T^R of n-variate polynomials of degree one over K, representing affine bijections s^{-1} and t^{-1} on K^n ;

S2-2. A partition $\pi = [n_1, \ldots, n_d]$ of the integer n such that

$$n = n_1 + \cdots + n_d \text{ and } 3 \leq n_1 \leq \cdots \leq n_d,$$

where $d \geq 1$ and $n_i = (2\ell_i + 1)2^{r_i}$ with $\ell_i \geq 1$ and $r_i \geq 0$;

S2-3. The π determines a bijection

$$\mu = [\mu_1, \ldots, \mu_d] : K^n \to K^{n_1} \times \cdots \times K^{n_d}$$

and projections

$$\mu_i : K^n \to K^{n_i}, \quad [x_0, \ldots, x_{n-1}] \mapsto [x_{a_i - n_i}, \ldots, x_{a_i - 1}],$$

where $a_i = \sum_{j=1}^{i} n_j$;

S2-4. A tuple $\Theta = [\theta_1, \ldots, \theta_d]$ of positive integers, where $\theta_i = b_i 2^{r_i}$ with $1 \leq b_i \leq \ell_i$;

S3. The structure of the public key : F represents the composite function $f : K^n \to K^n$, $f = t \circ \mu^{-1} \circ [\psi_{(n_1)}^{-1}, \ldots, \psi_{(n_d)}^{-1}] \circ [e_1, \ldots, e_d] \circ [\psi_{(n_1)}, \ldots, \psi_{(n_d)}] \circ \mu \circ s$, where

$$e_i : L_{(n_i)} \to L_{(n_i)}, \quad w_i \mapsto {w_i}^{1+q^{\theta_i}};$$

S4. The secret transformation algorithm SA, which outputs $\xi = SA(\Gamma, \eta)$:

step 1 {*Affine transformation*}: Evaluate T^R at $\eta \in K^n$ to obtain $v = T^R(\eta) \in K^n$;

step 2 {*Separation*}: Compute $\mu_1(v) \in K^{n_1}, \ldots, \mu_d(v) \in K^{n_d}$ from $v \in K^n$, i.e., split up a tuple of length n into d subtuples of lengths $n_1, \ldots, n_d$;

step 3 Execute the following steps for $i = 1$ to d :

(i-1) {*Decoding*}: According to the base of $L_{(n_i)}$ determined by the K-isomorphism $\psi_{(n_i)}$, translate the n_i-tuple $\mu_i(v)$ into an element $z_i = \psi_{(n_i)}(\mu_i(v))$ of $L_{(n_i)}$;

(i-2) $\{Powering\}$: Compute

$$w_i = z_i^{\overline{h_i}} \in L_{(n_i)}$$

from $z_i \in L_{(n_i)}$, where $\overline{h_i}$ is the multiplicative inverse of $h_i = 1 + q_i^{\theta_i}$ modulo $q_i^{n_i} - 1$;

(i-3) $\{Encoding\}$: Compute the vector-representation ${\psi_{(n_i)}}^{-1}(w_i) \in K^{n_i}$ of $w_i \in L_{(n_i)}$;

step 4 $\{Concatenation\}$: Compute

$$u = \mu^{-1}({\psi_{(n_1)}}^{-1}(w_1), \ldots, {\psi_{(n_d)}}^{-1}(w_d)) \in K^n,$$

i.e., concatenate d_i-tuples of lengths $n_1, \ldots, n_d$ into a tuple of length n ;

step 5 $\{Affine\ transformation\}$: Evaluate S^R at $u \in K^n$ to obtain $\xi = S^R(u) \in K^n$.

The validity of Definition 1 is checked and summarized as the following theorem.

Theorem 1. *For every appropriate pair $[F, \Gamma]$ of keys of C^* ,*

$$PA(F, SA(\Gamma, \eta)) = \eta, \quad \text{for any } \eta \in K^n,$$
$$SA(\Gamma, PA(F, \xi)) = \xi, \quad \text{for any } \xi \in K^n.$$

(Proof) See Chapter IV.

We can develop concrete algorithms for C^* and have the

Theorem 2. The size of a message block $S_{MB} = mn$ $[bit]$

The description length of

– a secret key = $D_{SK} \sim 2mn^2$ $[bit]$

– a public key = $D_{PK} \sim \frac{1}{2}mn^3$ $[bit]$

The circuit-size complexity (measured in the number of $GF(2)$-operations for one message block) of

- the secret key generation $= C_{SKG} = O(m^2n^2)$
- the public key generation $= C_{PKG} = O(m^2n^4)$
- the secret transformation $= C_{SA} = O(m^2n^2(m + \log n))$
  $(= O(m^2n(m+n))$ if $n = O(d)$)
- the public transformation $= C_{PA} = O(m^2n^3)$
  $(= O(m^2n^{2+\varepsilon}), (0 < \varepsilon \leq 1)$
  if transform n blocks at a time).

(Proof) See Chapter III .

Suppose that P is an n'-tuple of n-variate polynomials. Define a function τ_{up} by

$$\tau_{up}(P) = n' \cdot \binom{n + \deg(P)}{\deg(P)}.$$

It can be easily shown that the total number of nonzero terms of P, denoted by $\tau(P)$, is always less than or equal to $\tau_{up}(P)$.

For the security of C^* , the next theorem shows that it is practically infeasible for large n to extract the n-tuple G of n-variate polynomials representing the inverse of the function represented by the corresponding public key F.

Theorem 3. *The degree of n-variate n-tuple G satisfies :*

$$2^{m-1} = \frac{1}{2}q \leq \deg(G) \leq \frac{1}{2}\{(q-1)n_d + 1\}.$$

In particular, if n_d is odd and $gcd(\theta_d, n_d) = 1$, the most right inequality becomes an equality, and also a upper bound $\tau_{up}(G)$ of the number of terms in G satisfies

$$\tau_{up}(G) = n \cdot \binom{n + \deg(G)}{\deg(G)} > \{(\frac{\varepsilon n_d}{2n})(2^m - 1)\}^n \cdot (\frac{n}{2\pi})^{1/2}$$

where ε is the Napier's number $2.718\cdots$.

(Proof) See Chapter IV .

Small Example 2. The degree of G corresponding to F in Small Example 1 is equal to three because the F is based on $\pi = [3,5]$ and $\Theta = [1,1]$. The following is the exact description of the G.

$$\left\{\begin{aligned}
x_0 = G^{(0)}(y_0,\dots,y_7) &= y_0 + y_1 + y_4 + y_5 + y_6 + y_7 + y_0y_2 + y_0y_6 + y_0y_7 + y_1y_2 + y_1y_3 \\
&\quad + y_1y_4 + y_1y_5 + y_1y_6 + y_2y_3 + y_2y_6 + y_3y_4 + y_3y_6 + y_3y_7 + y_4y_6 + y_5y_6 + y_5y_7 \\
&\quad + y_0y_1y_4 + y_0y_1y_6 + y_0y_1y_7 + y_0y_3y_4 + y_0y_3y_6 + y_0y_3y_7 + y_0y_4y_5 + y_0y_4y_6 + y_0y_5y_6 \\
&\quad + y_0y_5y_7 + y_0y_6y_7 + y_1y_3y_4 + y_1y_3y_6 + y_1y_3y_7 + y_2y_3y_4 + y_2y_3y_6 + y_2y_3y_7 + y_2y_4y_5 \\
&\quad + y_2y_5y_6 + y_2y_5y_7 + y_3y_4y_5 + y_3y_4y_6 + y_3y_4y_7 + y_3y_5y_6 + y_3y_5y_7 + y_4y_5y_7 + y_5y_6y_7 \\
x_1 = G^{(1)}(y_0,\dots,y_7) &= 1 + y_0 + y_1 + y_4 + y_0y_2 + y_0y_3 + y_0y_5 + y_0y_6 + y_0y_7 + y_2y_4 + y_2y_5 \\
&\quad + y_2y_6 + y_2y_7 + y_3y_5 + y_3y_6 + y_4y_7 + y_5y_7 + y_6y_7 + y_0y_1y_2 + y_0y_1y_4 + y_0y_1y_6 + y_0y_2y_6 \\
&\quad + y_0y_4y_6 + y_1y_2y_5 + y_1y_3y_4 + y_1y_3y_6 + y_1y_3y_7 + y_1y_5y_7 + y_2y_5y_6 + y_3y_4y_6 + y_3y_6y_7 \\
&\quad + y_5y_6y_7 \\
x_2 = G^{(2)}(y_0,\dots,y_7) &= y_1 + y_2 + y_3 + y_5 + y_6 + y_0y_2 + y_1y_3 + y_1y_5 + y_1y_7 + y_2y_4 + y_2y_5 \\
&\quad + y_2y_7 + y_4y_5 + y_4y_7 + y_5y_6 + y_5y_7 + y_6y_7 + y_0y_1y_2 + y_0y_1y_7 + y_0y_2y_4 + y_0y_2y_7 + y_0y_3y_4 \\
&\quad + y_0y_3y_6 + y_0y_3y_7 + y_0y_4y_5 + y_0y_4y_7 + y_0y_5y_6 + y_0y_5y_7 + y_1y_2y_3 + y_1y_2y_4 + y_1y_2y_6 \\
&\quad + y_1y_2y_7 + y_1y_3y_4 + y_1y_3y_6 + y_1y_4y_5 + y_1y_4y_7 + y_1y_5y_6 + y_1y_5y_7 + y_1y_6y_7 + y_2y_3y_6 \\
&\quad + y_2y_4y_5 + y_2y_4y_6 + y_2y_5y_6 + y_2y_5y_7 + y_2y_6y_7 + y_3y_4y_5 + y_3y_4y_6 + y_3y_5y_6 + y_3y_5y_7 \\
&\quad + y_4y_5y_6 + y_4y_5y_7 + y_4y_6y_7 \\
x_3 = G^{(3)}(y_0,\dots,y_7) &= y_0 + y_1 + y_7 + y_0y_2 + y_1y_3 + y_1y_5 + y_1y_7 + y_2y_3 + y_2y_6 + y_2y_7 \\
&\quad + y_3y_4 + y_3y_5 + y_3y_6 + y_3y_7 + y_4y_5 + y_5y_6 + y_0y_1y_2 + y_0y_1y_7 + y_0y_2y_4 + y_0y_2y_7 + y_0y_3y_4 \\
&\quad + y_0y_3y_6 + y_0y_3y_7 + y_0y_4y_5 + y_0y_4y_7 + y_0y_5y_6 + y_0y_5y_7 + y_1y_2y_3 + y_1y_2y_4 + y_1y_2y_6 \\
&\quad + y_1y_2y_7 + y_1y_3y_4 + y_1y_3y_6 + y_1y_4y_5 + y_1y_4y_7 + y_1y_5y_6 + y_1y_5y_7 + y_1y_6y_7 + y_2y_3y_6 \\
&\quad + y_2y_4y_5 + y_2y_4y_6 + y_2y_5y_6 + y_2y_5y_7 + y_2y_6y_7 + y_3y_4y_5 + y_3y_4y_6 + y_3y_5y_6 + y_3y_5y_7 \\
&\quad + y_4y_5y_6 + y_4y_5y_7 + y_4y_6y_7 \\
x_4 = G^{(4)}(y_0,\dots,y_7) &= y_0 + y_2 + y_6 + y_0y_1 + y_0y_2 + y_0y_4 + y_0y_7 + y_1y_4 + y_1y_6 + y_1y_7 \\
&\quad + y_2y_4 + y_2y_5 + y_2y_6 + y_3y_4 + y_3y_6 + y_3y_7 + y_4y_5 + y_4y_7 + y_5y_6 + y_5y_7 + y_0y_1y_3 + y_0y_1y_5 \\
&\quad + y_0y_3y_4 + y_0y_3y_7 + y_0y_4y_5 + y_0y_5y_7 + y_1y_2y_3 + y_1y_2y_5 + y_1y_3y_4 + y_1y_3y_5 + y_1y_3y_6 \\
&\quad + y_1y_4y_5 + y_1y_5y_6 + y_2y_3y_6 + y_2y_5y_6 + y_3y_4y_5 + y_3y_4y_6 + y_3y_5y_7 + y_4y_5y_6 \\
x_5 = G^{(5)}(y_0,\dots,y_7) &= 1 + y_0 + y_2 + y_5 + y_6 + y_7 + y_0y_1 + y_0y_2 + y_0y_4 + y_0y_7 + y_1y_3 + y_1y_4 \\
&\quad + y_1y_6 + y_1y_7 + y_2y_3 + y_2y_5 + y_3y_6 + y_4y_6 + y_5y_7 + y_6y_7 + y_0y_1y_2 + y_0y_1y_7 + y_0y_2y_3 \\
&\quad + y_0y_2y_4 + y_0y_2y_5 + y_0y_2y_7 + y_0y_3y_4 + y_0y_3y_6 + y_0y_4y_5 + y_0y_4y_7 + y_0y_5y_6 + y_1y_2y_3 \\
&\quad + y_1y_3y_7 + y_2y_3y_4 + y_2y_3y_5 + y_2y_3y_7 + y_3y_4y_5 + y_3y_4y_7 + y_3y_5y_6 \\
x_6 = G^{(6)}(y_0,\dots,y_7) &= 1 + y_0 + y_2 + y_3 + y_4 + y_6 + y_0y_1 + y_0y_5 + y_0y_7 + y_1y_2 + y_2y_5 + y_2y_6 \\
&\quad + y_2y_7 + y_3y_4 + y_3y_5 + y_3y_6 + y_3y_7 + y_4y_5 + y_5y_6 + y_0y_1y_2 + y_0y_1y_3 + y_0y_1y_4 + y_0y_1y_5 \\
&\quad + y_0y_1y_6 + y_0y_2y_4 + y_0y_2y_7 + y_0y_3y_4 + y_0y_3y_7 + y_0y_4y_5 + y_0y_4y_6 + y_0y_4y_7 + y_0y_5y_7 \\
&\quad + y_0y_6y_7 + y_1y_2y_4 + y_1y_2y_5 + y_1y_2y_6 + y_1y_2y_7 + y_1y_3y_4 + y_1y_3y_5 + y_1y_3y_6 + y_1y_3y_7 \\
&\quad + y_1y_4y_7 + y_1y_5y_7 + y_1y_6y_7 + y_2y_3y_4 + y_2y_3y_6 + y_2y_3y_7 + y_2y_4y_6 + y_2y_5y_6 + y_2y_6y_7 \\
&\quad + y_3y_4y_5 + y_3y_4y_6 + y_3y_4y_7 + y_3y_5y_7 + y_4y_6y_7 + y_5y_6y_7 \\
x_7 = G^{(7)}(y_0,\dots,y_7) &= 1 + y_1 + y_4 + y_6 + y_7 + y_0y_1 + y_0y_2 + y_0y_4 + y_0y_5 + y_1y_2 + y_2y_4 + y_3y_7 \\
&\quad + y_4y_6 + y_4y_7 + y_5y_6 + y_5y_7 + y_6y_7 + y_0y_1y_3 + y_0y_1y_5 + y_0y_2y_4 + y_0y_2y_6 + y_0y_2y_7 \\
&\quad + y_0y_3y_4 + y_0y_3y_7 + y_0y_4y_5 + y_0y_4y_7 + y_0y_5y_7 + y_0y_6y_7 + y_1y_2y_4 + y_1y_2y_6 + y_1y_2y_7 \\
&\quad + y_1y_3y_5 + y_1y_4y_7 + y_1y_6y_7 + y_2y_3y_4 + y_2y_3y_6 + y_2y_3y_7 + y_2y_4y_6 + y_2y_6y_7 + y_3y_4y_5 \\
&\quad + y_3y_4y_7 + y_3y_5y_7 + y_3y_6y_7 + y_4y_6y_7
\end{aligned}\right.$$

Besides the above aspect, we must discuss the complexity of deducing a secret key by decomposing the corresponding public key; the period of the public transformation which reflects the robustness of the system against the iteratively-transforming-attack; the relation between bit-security and block-security, etc.

For small values of the parameters m and n, we have some experimental results showing that there seems to be no apparent clues to reduce the complexities of the above mentioned atacks. However, more advanced theories should be necessary to confirm this circumstantial evidence.

In our present point of view, if the parameters are set to be $1 \leq m \leq 32$, $32 \leq n \leq 64$, and $64 \leq mn$, then C^* can achieve both high security and great realizability.

III. ALGORITHMS FOR C^* AND THEIR COMPLEXITY

III -1. Secret Key and Its Generation

As defined in Definition 1, a secret key for C^* consists of four parts: two n-tuples of linear n-variate polynomials S^R and T^R, a partition $\pi = [n_1, \ldots, n_d]$ of n and a tuple of integers $\Theta = [\theta_1, \ldots, \theta_d]$.

First, we consider S^R and T^R. Let B represent either of them. B can be represented by an n-tuple B_∞ over K and an n-dimensional square matrix B_ℓ as follows:

$$B(x_0, \ldots, x_{n-1}) = B_\infty + [x_0, \ldots, x_{n-1}]B_\ell.$$

B is bijective iff the matrix B_ℓ is nonsingular.

As there are a great many nonsingular matrices, B_ℓ can be found using the method of trial and error. However, it will be shown in Section III -3 that to generate a public key, we have to solve the following linear system in $x_0, \ldots, x_{n-1}$:

$$[y_0, \ldots, y_{n-1}] = B(x_0, \ldots, x_{n-1}). \tag{1}$$

Hence we can use an excellent method — the LDU decomposition method. That is, we can first select an n-dimensional lower triangular matrix $\mathbf{L}$ over K whose diagonal components are all 1, a non-zero n-dimensional diagonal matrix $\mathbf{D}$ over K, and an n-dimensional upper triangular matrix

$\mathbf{U}$ over K whose diagonal components are all 1, then find the product of them

$$B_\ell = \mathbf{LDU}.$$

Apparently, B_ℓ is nonsingular, since $\mathbf{L}, \mathbf{D}$ and $\mathbf{U}$ are all nonsingular. Of course, there are other nonsingular matrices not expressible by the above formula, but that part is very small. Using $\mathbf{L}, \mathbf{D}$ and $\mathbf{U}$ but not B_ℓ, solving the system (1) becomes fairly convenient.

Obviously, it requires $mn(n+1)$ $[bit]$ to describe B. Further, it requires $2\sum_{i=1}^{d} \log n_i [bit]$ to describe π and Θ which cannot exceed n $[bit]$. Thus, we have the following estimations:

$$\begin{aligned} D_{SK}\{&\text{the description length of secret key of } C^*\} \\ &= (2m(n+1)+1)n \ [bit] \\ &\sim 2mn^2 \ [bit], \\ C_{SKG}\{&\text{the circuit} - \text{size complexity of secret key generation of } C^*\} \\ &= O(m^2n^2) \ [GF(2) - \text{operation}]. \end{aligned}$$

III -2. The Secret Transformation Algorithm

The secret transformation algorithm SA consists of (step 1)~(step 5) outlined in Definition 1. The running time of (step 1) and (step 5) performing affine transformations is clearly $O(m^2n^2)$ $[GF(2) - \text{operation}]$. As compared with the other steps, the running time of (step 2), (step 3-i-1), (step 3-i-3) and (step 4) can be neglected. Now what remain to be investigated are only the concrete algorithm which performs powering in step (step 3-i-2), and its complexity. Taking advantage of features of $\overline{h}_i$, this section constructs an efficient algorithm for the powering.

First, we have the following theorem.

Theorem 4. *For integers m, q, ℓ, n, b, and θ satisfying $m > 0, q = 2^m, \ell > 0, r \geq 0, n = (2\ell+1)2^r, 0 < b \leq \ell, \theta = b2^r$, the integer $h = 1 + q^\theta$ possesses a multiplicative inverse element $\overline{h}$ modulo $(q^n - 1)$, which can be expressed as*

$$\overline{h} \equiv \{(\sum_{i=0}^{m-1} 2^i)(\sum_{j=0}^{\theta-1} q^j)(\sum_{k=0}^{\ell-1} q^{2\theta k}) + q^{2\theta\ell}\} \cdot q^{\theta-1} \cdot 2^{m-1} \pmod{q^n - 1}. (2)$$

Proof: We notice that

$$(\sum_{i=0}^{m-1} 2^i)(\sum_{j=0}^{\theta-1} q^j) = (q-1)(\sum_{j=0}^{\theta-1} q^j) = q^\theta - 1$$

and that $q^{\theta-1} \cdot 2^{m-1} = \frac{1}{2}q^\theta$. Thus we have

$$\begin{aligned} h \cdot (\text{RHS of } (2)) &= \{(1+q^\theta)(q^\theta - 1)(\sum_{k=0}^{\ell-1} q^{2\theta k}) + (1+q^\theta)q^{2\theta\ell}\}(\frac{1}{2}q^\theta) \\ &= \{(q^{2\theta} - 1)(\sum_{k=0}^{\ell-1} q^{2\theta k}) + q^{2\theta\ell} + q^{(2\ell+1)\theta}\}(\frac{1}{2}q^\theta) \\ &= (q^{2\theta\ell} - 1 + q^{2\theta\ell} + q^{nb})(\frac{1}{2}q^\theta) \\ &\equiv (2q^{2\theta\ell} - 1 + 1)(\frac{1}{2}q^\theta) \pmod{q^n - 1} \\ &= q^{(2\ell+1)\theta} \\ &= q^{nb} \\ &\equiv 1 \pmod{q^n - 1}. \quad ♣ \end{aligned}$$

From this theorem we see that the $\overline{h}$th power of an element of a finite field L of order q^n can be computed from the $(\sum_{i=0}^{a-1} b^i)$th and the b^ath powers of the element for some a and b.

Let us consider evaluating the $(\sum_{i=0}^{a-1} b^i)$th power. For example, we can use the fact that

$$\sum_{i=0}^{6} b^i = ((b^2+1)b^2+1)(b+1)b+1 \tag{3}$$

to compute z, the $\sum_{i=0}^{6} b^i$th power of x, by the following algorithm:

```
(step1)  $y \longleftarrow x$ ;
(step2)  $y \longleftarrow y^{b^2} \cdot y$ ;
(step3)  $y \longleftarrow y^{b^2} \cdot x$ ;
(step4)  $y \longleftarrow y^b \cdot y$ ;
(step5)  $y \longleftarrow y^b \cdot x$ ;
(step6)  $z \longleftarrow y$ .
```

This algorithm requires 4 multiplications and 4 evaluations of the b^kth power (where k is a suitable positive integer). The latter operation requires about 6 times b-th powering.

Similarly, for general $\sum_{i=0}^{a-1} b^i$, to evaluate the $(\sum_{i=0}^{a-1} b^i)$th power can be completed by using a formula like (3). The complexity is estimated as follows.

Theorem 5. *For two positive integers a and b, evaluating the $(\sum_{i=0}^{a-1} b^i)$th power can be accomplished in*

(ζ) $\lfloor \log_2 a \rfloor + W_2(a) - 1$ *times multiplications ;*

(η) $\lfloor \log_2 a \rfloor + W_2(a) - 1$ *times evaluation of the b^kth power,*

where k is a suitable positive integer and where $W_2(a)$ denotes the 2-weight of a defined by

$$W_2(a) = \sum \{a_j | j = 0, 1, \cdots\},$$

when the binary representation of a is

$$a = \sum \{2^j \cdot a_j | 0 \leq a_j < 2, \quad j = 0, 1, \cdots\}.$$

Furthermore, if evaluating the b^kth power is done by evaluating iteratively the bth power, then (η) can be expressed as

(η') $(a - 1)$ *times evaluation of the bth power.*

Proof(sketch): For a general $(\sum_{i=0}^{a-1} b^i)$, we form the corresponding formula like (3). Counting the number of "+" appearing in the right hand side of the formula, we get (ζ) and (η); summing the superscripts of b, we get (η'). ♣

Corollary 1. *For positive integers a and b, the complexity of evaluating the $(\sum_{i=0}^{a-1} b^i)$th power is estimated as "$O(\log a)$ times multiplication" , if the complexity of evaluating the bth power can be neglected as compared to that of multiplication.*

It is known[9] that, for the n-degree extension field L of a finite field K of order q, there always exists a base of L over K which takes the form of $[\beta, \beta^q, \beta^{q^2}, \ldots, \beta^{q^{n-1}}]$ (called a normal base). Let $V(x) = [x_0, \ldots, x_{n-1}] \in K^n$ denote the (vector) representation of an element x of L by a normal base $[\beta, \beta^q, \beta^{q^2}, \ldots, \beta^{q^{n-1}}]$, i.e., x is expressed as $x = \sum_{i=0}^{n-1} x_i \beta^{q^i}$. Now, for any integer k, we have

$$\begin{aligned} V(x^{q^k}) &= [x_{(0-k) \bmod n}, x_{(1-k) \bmod n}, \ldots, x_{(n-1-k) \bmod n}] \quad (4) \\ &= CS_k(V(x)) \end{aligned}$$

since $x_i^q = x_i$, where $CS_k(V(x))$ is a bijection on K^n, and represents the k-step (right) cyclic shift operation.

By the use of this well-known fact, we see that the complexity of evaluating the q^kth power of an element of L, can be neglected as compared to the complexity of the L-multiplication (the multiplication of two elements over L), if elements of L are represented by using a normal base of L over K.

Assembling all the above results, we get an algorithm for evaluating the $\overline{h}$th power over the field L of order q^n, where q, n, $\overline{h}$ satisfy the conditions stated in Theorem 4.

[HPA : algorithm for evaluating the $\overline{h}$th power]

PREREQUISITE: Each element of L is given in the form of a vector representation by a normal base of L over K.

PROCEDURE (Outline): Evaluating the $\overline{h}$th power according to (2). Notice that evaluating the $\sum_{i=0}^{m-1} 2^i$th, the $\sum_{j=0}^{\theta-1} q^j$th and the $\sum_{k=0}^{\ell-1} q^{2\theta k}$th powers, is decomposed into the L-multiplication and evaluating the 2^γth and the q^δth powers of elements of L by using formulae like (3). Also, all the evaluations of the q^ϵth power are performed by cyclic shifts to the right.

The complexity of the algorithm HPA is estimated in the following theorem.

Theorem 6. *For* HPA, *$O(m + \log n)$ times L-multiplication are sufficient for evaluating the $\overline{h}$th power of an element of L. And hence, the circuit-size complexity of* HPA *is*

$$O(m^2 n^2 (m + \log n)) \qquad [GF(2) - \text{operation}].$$

Proof: From Corollary 1, we know that evaluating the $\sum_{j=0}^{\theta-1} q^j$th and the $\sum_{k=0}^{\ell-1} q^{2\theta k}$th powers can be performed in $O(\log\theta)$ and $O(\log\ell)$, respectively, times L-multiplication. Since $\theta, \ell < n/2$, the summation of them is $O(\log n)$. And also we know, from Theorem 5, that evaluating the $\sum_{i=0}^{m-1} 2^i$th power can be performed in at most $m - 1 + \lfloor\log m\rfloor + W_2(m) - 1 = O(m)$ times L-multiplication. Further, evaluating $q^{2\theta k}$th and the $q^{\theta-1}$th powers can be done only by cyclic shifts, hence the complexities of them can be neglected. Now, evaluating the 2^{m-1}th power can be accomplished in $(m-1)$ times multiplication. Summing all the the above terms, we get the first half of the theorem. The second half of the theorem is obvious, since the L-multiplication can be done in $O(m^2n^2)$ times operations over $GF(2)$. ♣

Thus, when the algorithm HPA is used in evaluating power, the total circuit-size complexity C_{SA} of the secret transformation algorithm is estimated by

$$O(m^2n^2) + \sum_{i=1}^{d} O(m^2n_i^2(m + \log n_i)) \qquad [GF(2) - \text{operation}].$$

The above estimation can be further condensed to

$$C_{SA} = O(m^2n^2(m + \log n)) \qquad [GF(2) - \text{operation}].$$

In particular, if there is a constant c_0 independent from n such that $n_i < c_0$, i.e., if $n = O(d)$, then it holds $\sum_{i=1}^{d} O(m^2n_i^2(m+\log n_i)) = O(m^3n)$, which implies that the circuit-size complexity of the secret transformation algorithm can be estimated by

$$C_{SA} = O(m^2n(m + n)) \qquad [GF(2) - \text{operation}].$$

III -3. Public Key and Its Generation

A public key F of C^* is an n-tuple of n-variate polynomials over K. So obviously, we have

$$\begin{aligned} D_{PK} &= m\tau_{up}(F) \quad [bit] \\ &= \frac{1}{2}mn(n+1)(n+2) \quad [bit] \\ &\sim \frac{1}{2}mn^3 \quad [bit] \end{aligned}$$

for the description length D_{PK} of a public key of C^* .

Next, we consider how to generate a public key F. F can be expressed by n-tuples $F_\infty, F_i, F_{ii}, F_{ij} \in K^n$ as

$$F(x_0, \ldots, x_{n-1}) = F_\infty + \sum_{i=0}^{n-1} F_i x_i + \sum_{i=0}^{n-1} F_{ii} x_i^2 + \sum_{0 \leq i < j < n} F_{ij} x_i x_j \quad (5)$$

where $F_{ii} = 0$ when $m = 1$. Thus, we can first compute, according to the definition of the public transformation, values of F at the points corresponding to several elements of K^n, then from these values, find $F_\infty, F_i, F_{ii}, F_{ij}$ by the use of the interpolation method, and finally, generate the desired F.

Now suppose that $\eta_\infty \in K^n$ is a 0 vector, $\eta_i \in K^n$ a vector whose ith $(0 \leq i < n)$ coodinate is 1 but all of the others are 0, and $\eta_{ij} \in K^n$ a vector whose ith and jth $(0 \leq i < j < n)$ coodinates are 1 but all of the others are 0. When $m \geq 2$, we have

$$\begin{aligned} F(\eta_\infty) &= F_\infty, \\ F(\eta_i) &= F_\infty + F_i + F_{ii}, \\ F(a\eta_i) &= F_\infty + aF_i + a^2 F_{ii} \quad (a \in K, a \neq 0, 1), \\ F(\eta_{ij}) &= F_\infty + F_i + F_j + F_{ii} + F_{jj} + F_{ij}. \end{aligned}$$

When $m = 1$, we have

$$\begin{aligned} F(\eta_\infty) &= F_\infty, \\ F(\eta_i) &= F_\infty + F_i, \\ F(\eta_{ij}) &= F_\infty + F_i + F_j + F_{ij}. \end{aligned}$$

Hence, the n-tuple F of n-vatiate polynomials can be computed from

$$\left.\begin{aligned} F_\infty &= F(\eta_\infty), \\ F_{ii} &= \begin{cases} \dfrac{1}{a(a-1)}(F(a\eta_i) - aF(\eta_i) + (1-a)F_\infty); & m \geq 2 \\ 0: \quad m = 1, \end{cases} \\ F_i &= F(\eta_i) - F_{ii} - F_\infty, \\ F_{ij} &= F(\eta_{ij}) - F(\eta_i) - F(\eta_j) + F(\eta_\infty). \end{aligned}\right\} \quad (6)$$

Applying this method, we have the following algorithm :

```
[PKG : algorithm for generating a public key ]
    (step 1){$Evaluating\ \gamma = F(\eta) \in K^n\ at\ \eta = \eta_\infty, \eta_j, a\eta_j, \eta_{jk}$ }
      (step 1-1) : Compute $\omega \in K^n$ satisfying $S^R(\omega) = \eta$ ;
      (step 1-2) : Find $\mu_i(\omega) \in K^{n_i}$ $(1 \leq i \leq d)$;
      (step 1-3) : Execute the following steps for $i = 1$ to $d$ :
        (step 1-3-i-1) : Find $\omega_i = \psi_{(n_i)}(\mu_i(\omega)) \in L_{(n_i)}$;
        (step 1-3-i-2) : Compute $z_i = \omega_i^{h_i} \in L_{(n_i)}$;
        (step 1-3-i-3) : Find $\psi^{-1}_{(n_i)}(z_i) \in K^{n_i}$;
      (step 1-4) : Find $xi = \mu^{-1}(\psi^{-1}_{(n_1)}(z_1), \ldots, \psi^{-1}_{(n_d)}(z_d)) \in K^n$;
      (step 1-5) : Find $\gamma \in K^n$ satisfying $T^R(\gamma) = \xi$;
    (step 2) : Find $F_\infty, F_i, F_{ii}, F_{ij}$ according to (6).
```

Using the matrices **L**, **D** and **U**, based on which S^T and T^R were computed in Section III -1, (step 1-1) and (step 1-5) can be executed in $O(n^2)$ K-operation. According to Theorem 5, (step 1-3-i-2) can be executed in $O(m^2 n_i^2)[GF(2) - \text{operation}]$. Notice that the complexities of the other steps can be neglected as compared to these, and there are totally $\binom{n+2}{2}$ points η to be used, we can estimate the complxity of (step 1) by

$$\binom{n+2}{2}\{2O(m^2n^2) + \sum_{i=1}^{d} O(m^2 n_i^2)\} = O(m^2 n^4) \qquad [GF(2) - \text{operation}].$$

From (6), the complexity of (step 2) is estimated as

$$n \cdot O(m^2 n) + n \cdot O(mn) + \binom{n}{2} \cdot O(mn) = O(mn^2(m+n)) \quad [GF(2) - \text{operation}].$$

Thus we conclude that

$$C_{PKG}\{\text{the circuit} - \text{size complexity of public key generation of } C^*\} \\ = O(m^2 n^4) \quad [GF(2) - \text{operation}].$$

III -4. Public Transformation Algorithm

As noted in Definition 1, the public transformation algorithm PA evaluates the polynomial tuple F at points of K^n. Let

$$\tilde{x} = [x_0, \ldots, x_{n-1}, x_0^2, \ldots, x_{n-1}^2, x_0x_1, \ldots, x_{n-2}x_{n-1}]$$

be a vector corresponding to $x = [x_0, \ldots, x_{n-1}]$, and

$$\tilde{F} = [F_0, \ldots, F_{n-1}, F_{00}, \ldots, F_{n-1,n-1}, F_{01}, \ldots, F_{n-2,n-1}]^T$$

be a $\frac{1}{2}n(n+3) \times n$ matrix. Using $\tilde{x}$ and $\tilde{F}$, we can rewrite (5) as

$$F(x) = F_\infty + \tilde{x}\tilde{F}.$$

So, we can first find $\tilde{x}$, then find $F(x)$ to perform the public transformation. This complexity is

$$\begin{aligned} C_{PA} &= \frac{1}{2}n(n+1) \cdot O(m^2) + n \cdot \frac{1}{2}(n+1)(n+2) \cdot O(m^2) \\ &= O(m^2n^3) \qquad\qquad [GF(2) - \text{operation}]. \end{aligned}$$

Furthermore, when performing public transformation on n message blocks $x^{(0)}, \ldots, x^{(n-1)}$ in parallel, we can do it by computing $A_0 + \tilde{X}\tilde{F}$ according to an $n \times \frac{1}{2}n(n+3)$ matrix $\tilde{X} = [\tilde{x}^{(0)}, \ldots, \tilde{x}^{(n-1)}]^T$ and an $n \times n$ matrix $A_0 = [F_\infty, \ldots, F_\infty]^T$. $A_0 + \tilde{X}\tilde{F}$ can be rewitten into

$$A_0 + X_1A_1 + \cdots + X_{n+3}A_{n+3}$$

where X_i and A_i are $n \times n$ matrices and satisfies $\tilde{X} = [X_1, \ldots, X_{n+3}]$ and $\tilde{F} = [A_1, \ldots, A_{n+3}]^T$, respectively. Here, we can multiply two matrices in $O(n^{2+\varepsilon})$ [K-operation] $(0 < \varepsilon \leq 1)$ by the use of various, say Strassen's, *divide and conqure methods.* Thus, in this case, the circuit-size complexity of public transformation for one block is

$$\{(n+3) \cdot O(n^{2+\varepsilon}) \cdot O(m^2)\}/n = O(m^2n^{2+\varepsilon}) \ [GF(2) - \text{operation}].$$

III -5. Collection of Main Results

Theorem 2 can be directly proved by the results of the above four sections.

Corollary 2. *For C^* with $m \sim n$ and $mn = N$, the parameters in Theorem 2 become :*

$$\{S_{MB} = N\} \quad \left\{ \begin{array}{c} D_{SK} \sim 2N^{1.5} \\ C_{SKG} = O(N^2) \\ C_{SA} = O(N^{2.5}) \\ (\text{or} = O(N^2)) \end{array} \right\} \quad \left\{ \begin{array}{c} D_{PK} \sim \frac{1}{2}N^2 \\ C_{PKG} = O(N^3) \\ C_{PA} = O(N^{2.5}) \\ (\text{or} = O(N^{2+\epsilon})) \end{array} \right\}.$$

Now we briefly compare C^* with the RSA cryptosystem[10]. For the RSA system, the complexities of secret transformation and public transformation are both $O(N^3)$ for a block of size N. When a particular secret key or a public key is selected, the corresponding complexity can be reduced to less than $O(N^3)$. However, it seems that, in general, we have no way to reduce both of them. As opposed to the above fact, the order of the complexity of public transformation of C^* is much lower than that of the RSA system. Also, for the RSA system, public and secret keys connot be generated if an integer with certain particular properties is not found. For C^* , keys can be easily genarated.

The description length of a key for C^* is greater than those of previous systems with the same block size. However, this is not always a demerit because the total number of usable keys of C^* is larger than that of those. Further, the large description length will not be a serious problem, if public keys are kept by the corresponding owner after they are certificated by the manager of the system or network, and when necessary, sent to other ones with the certificates.

III -6. Implementation—Primary Results

Using a 32-bit microprocessor $MC68020$ (16.67 MHz) on a SONY NEWS UNIX workstation with programs written in the "C " language, our first implementation confirms that algorithms SA and PA run at least 100 $Kbps$ for $m = 8$ and $n = 32$. Since these programs are not optimized, we may expect that C^* can run much faster in the same environment.

Besides this , we also have been implementing C^* using multiple transputers ($T414, T800$) with *occam* programs, and verifying high performance. Detailed results will appear in another paper.

IV. A THEORY OF POLYNOMIAL-TUPLE ASYMMETRIC CRYPTOSYSTEMS

In this chapter we discuss why we have stated C^* as Definition 1 and prove Theorem 1 and Theorem 3.

IV -0. Preliminaries

Basic concepts and notations used in this chapter are sketched in the following.

Finite Fields[9]

Let p be a prime integer, m and n positive integers, and $q = p^m$. Fix a finite field K of order q (i.e., with q elements). Denote by K^n the n-dimensional vector space over K, each element of which is an n-tuple over K. Determine an n-degree extension field L of K. L contains q^n elements. When L is taken as an n-dimensional vector space over K, L is isomorphic to K^n. The isomorphism between L and K^n will be denoted by a bijection $\psi : K^n \to L$.

Polynomial Representations of Functions

Denote by $L[u]$ the polynomial ring over L in indeterminate u, and by $(P(u))$ the ideal generated by a polynomial $P(u) \in L[u]$. As shown in [11], any function $f_1 : L \to L$ can be represented by a univariate polynomial $E(u) \in L[u]$, where $E(u)$ is uniquely determined in the residue class ring $L[u]/(u^{q^n} - u)$ (i.e., mod$(u^{q^n} - u)$ is applied). In other words, we always have $f_1(\xi) = E(\xi)$ for every $\xi \in L$, and furthermore, there is just one such $E(u)$ which has no terms divisible by u^{q^n}. Such an $E(u)$ is called the univariate polynomial representation of f_1 over L, and denoted by $[\![f_1]\!]$.

Similarly, functions $f_2 : L \to K^n, f_3 : K^n \to L$, and $f_4 : K^n \to K^n$ can be uniquely represented by a tuple of polynomials over L in indeterminate u mod$(u^{q^n} - u)$, a polynomial over L in indeterminates $x_0, \ldots, x_{n-1}$ mod $(x_0^q - x_0, \ldots, x_{n-1}^q - x_{n-1})$, and a tuple of polynomials over K in indeterminates $x_0, \ldots, x_{n-1}$ mod $(x_0^q - x_0, \ldots, x_{n-1}^q - x_{n-1})$, respectively. These items are called the univariate polynomial n-tuple representation of f_2 over L, the n-variate polynomial representation of f_3 over L, and the n-variate polynomial n-tuple representation of f_4 over K, and denoted by $[\![f_2]\!], [\![f_3]\!]$ and $[\![f_4]\!]$, respectively.

Functions Represented by Algorithms

Polynomials or tuples of polynomials can be considered to be a kind of *algorithms.* In general, there are two sets I and J with related to an algorithm A. When A outputs $\eta \in J$ on input $\xi \in I$, we say A represents a function $I \to J, \xi \mapsto \eta$, and denote the function by $\langle A \rangle$. For example, since the polynomial representation $[\![f_1]\!]$ of the function f_1 is considered to be an algorithm, it is apparent that $\langle [\![f_1]\!] \rangle = f_1$.

Functions on Integers

Let a be an integer greater than 1, i a nonnegative integer. Denote the a-ary representation of i by

$$i = \sum \{a^j \cdot i_j | 0 \le i_j < a, \quad j = 0, 1, \cdots\}.$$

We define a function W_a on the nonnegative integers as follows:

$$W_a(i) = \sum \{i_j | j = 0, 1, \cdots\}.$$

$W_a(i)$ is called the a-weight of i, which has the following properties:

(W1) If $s \ge 0, t \ge 0$ and $0 \le s + t < a$, then

$$W_a(s+t) = W_a(s) + W_a(t).$$

(W2) If $0 \le t < a$, then

$$\begin{aligned} W_a(as+t) &= W_a(as) + W_a(t) \\ &= W_a(s) + W_a(t). \end{aligned}$$

(W3) If $s \ge 0, t \ge 0$ and $s + t = a^n - 1$, then

$$n(a-1) = W_a(s+t) = W_a(s) + W_a(t).$$

Also, we define a function R_a from the positive integers to the nonnegative integers as follows:

$$R_a(i) = \max\{j \ge 0 | i \text{ is divisible by } a^j\}.$$

$R_a(i)$ is called the a-rank of i, which has the following properties:

(R1) $R_a(i)$ is equal to the number of consecutive 0's appearing in the least significant digits of the a-ary representation of the positive integer i.

(R2) If a is a prime and $s > 0$ and $t > 0$, then $R_a(s \cdot t) = R_a(s) + R_a(t)$.

Functions from Polynomials to Integers

For a univariate polynomial $E(u) = E_0 + E_1 u + E_2 u^2 + \cdots + E_d u^d$, the exponential a-weight $wt_a(E)$ of E is defined by

$$wt_a(E) = \begin{cases} \max\{W_a(i) | E_i \neq 0\}; & E \neq 0, \\ -\infty; & E = 0. \end{cases}$$

Besides this, we use the notations $deg(P)$, $\tau(P)$, and $\tau_{up}(P)$ for polynomial-tuple P as defined in Chapter I.

IV -1. Multivariate Equations and Cryptosystems

Imagine that we are to realize a public-key signature scheme, when given an asymmetric cryptosystem with multivariate polynomial-tuples as public keys. Finding the valid signature x with respect to a message M and a public key F can be rephrased as solving the equation $F(x) = M$ for x given F and M. The essential idea behind the present research is that we can employ a system of multivariate algebraic equations as the equation $F(x) = M$. The grounds for it are that, in general, as briefly introduced in Chapter I, it is an extremely difficult problem to solve systems of multivariate algebraic equations. Of course, when given hints about a system, say some information on the structure of F, one may be able to to solve the system quickly.

In the rest of this chapter, we will aim at constructing a system of multivariate algebraic equations $F(x) = M$. The system corresponds to an asymmetric cryptosystem supporting both authenticity and confidentiality, so we cannot say the system is a completely general one. But it should not be easy to get any hint on effectively solving the system of equations, i.e., the system should possess no apparent features. In a sense, the system should be a nearly random one.

IV -2. From Univariate Polynomials Into Tuples of Multivariate Polynomials

For our purposes, we require that the above tuple F of multivariate polynomials represents a bijection, and that the equation $F(x) = M$ can be readily solved when given some knowledge on it. Hence, we take the following approach[4] : We begin our discussion by thinking about univariate polynomials. Coping with such polynomials is relatively easy. Then we transform them into multivariate ones. Several aspects have to be considered : (1) Tuples of multivariate polynomials must be made as random as possible; (2) It should be easy to estimate the size, and the likes, of the resulting multivariate polynomial-tuples from the basic univariate polynomials.

Here is an idea. Following the ways of thinking on the algorithm composition method proposed in [5], we consider a function $f : K^n \rightarrow K^n$ expressed as follows (K is a finite field of order $q = p^m$ with prime p):

$$f = t \circ g \circ s \tag{7}$$

$$g = \mu^{-1} \circ [\psi_1^{-1} \circ e_1 \circ \psi_1 \circ \mu_1, \cdots, \psi_d^{-1} \circ e_d \circ \psi_d \circ \mu_d] \tag{8}$$

where s and t are affine bijections on K^n, n is a positive integer which can be partitioned into d positive integers satisfying $n = n_1 + n_2 + \cdots + n_d$, and L_i is an n_i-degree extension field of the field K. ψ_i is an isomorphism from K^{n_i} to L_i, and e_i a bijection on L_i. Further, $\mu_i : K^n \rightarrow K^{n_i}$ is a projection which maps $[x_0, \ldots, x_{n-1}] \in K^n$ to $[x_{\sum_{j=1}^{i-1} n_j}, \ldots, x_{(\sum_{j=1}^{i} n_j)-1}] \in K^{n_i}$, and $\mu : K^n \rightarrow K^{n_1} \times \cdots \times K^{n_d}$ is a bijection determined by $\mu = [\mu_1, \ldots, \mu_d]$.

Apparently, the function f is a bijection. Now we establish an asymmetric cryptosystem which uses f as a public transformation.

Definition 2. Let K^n be the set of message blocks. The following system constitutes an asymmetric cryptosystem. The system is constructed by designating

(1) $[\![f]\!]$, an n-tuple of n-variate polynomials over K, as a public key;

(2) $[\![t^{-1}]\!], [\![e_1^{-1}]\!], \ldots, [\![e_j^{-1}]\!]$ and $[\![s^{-1}]\!]$ as a secret key;

(3) the evaluation of $[\![f]\!]$ as the public transformation algorithm;

(4) the operations series in the following order as the secret transformation algorithm:

(a) the evaluation of $[\![t^{-1}]\!]$,

(b) the projections due to μ_i,

(c) the transformations due to ψ_i,

(d) the evaluations of $[\![e_i^{-1}]\!]$,

(e) the transformations due to ψ_i^{-1},

(f) the concatenation due to μ, and

(g) the evaluation of $[\![s^{-1}]\!]$.

This asymmetric cryptosystem will be called C_0^* for short.

IV -3. Degree of A Tuple of Multivariate Polynomials

Now, the size of public key and the complexity of public transformation of C_0^* can be estimated by the following formulae:

$$\{\text{ The description length of a public key of } C_0^* \} = O(\tau([\![f]\!]) \log_2 p)[bit].$$

$$\{\text{ The complexity of a public transformation of } C_0^* \}$$
$$= O(\tau([\![f]\!])m^2)\ [GF(2) - \text{operation}].$$

Clearly, both the descreption length of a public key and the complexity of a public transformation are increasing functions of $\tau([\![f]\!])$ — the number of terms in the n-tuple $[\![f]\!]$ (the public key) of n-variate polynomials. From the equations (7) and (8), we can see that $[\![f]\!]$ is hardly sparse, but dense in most cases. Thus, decreasing deg $([\![f]\!])$ which dominates the upper bound $\tau_{up}([\![f]\!])$ of $\tau([\![f]\!])$, is strongly related to reducing the description length of a public key and the complexity of the public transformation.

Similarly, it is also true that in most cases, the polynomial representation $[\![f^{-1}]\!]$ of a secret transformation f^{-1} of C_0^* is dense. Therefore, increasing deg $([\![f^{-1}]\!])$ which dominates $\tau_{up}([\![f^{-1}]\!])$ is related to raising the number of terms in $\tau([\![f^{-1}]\!])$, and also related to raising tremendously the complexity of extracting the secret key $[\![f^{-1}]\!]$ from the public key $[\![f]\!]$ by the use of the symbolic computation, the interpolation, or other methods for solving algebraic equations.

First, turn our attention to a basic theorem.

Theorem 7. *Let s and t be any two affine functions on the vector space K^n, E denote the set of all functions on the finite field L. We have the following (i), (ii) and (iii) :*

(i) For any $e \in E$,

$$[\![e]\!] = constant \Longrightarrow [\![t \circ \psi^{-1} \circ e \circ \psi \circ s]\!] = constant.$$

(ii) For any $e \in E$,

$$[\![e]\!] \neq 0 \Longrightarrow \deg([\![t \circ \psi^{-1} \circ e \circ \psi \circ s]\!]) \leq wt_q([\![e]\!]).$$

(iii) If and only if both s and t are bijections, the following holds for all $e \in E$

$$[\![e]\!] \neq constant \Longrightarrow \deg([\![t \circ \psi^{-1} \circ e \circ \psi \circ s]\!]) = wt_q([\![e]\!]).$$

Proof (sketch): Proving this theorem is not difficult but wastes pages. So, we mention here only that the proof for general q can be readily obtained from that for the case $q = 2$, which is described in [12]. ♣

Using Theorem 7, we can compute the degree of the multivariate polynomial tuple $[\![f]\!]$ from the exponential q-weights of univariate polynomials $[\![e_1]\!], \ldots, [\![e_d]\!]$. The computing method is described in the following theorem.

Theorem 8. *For the bijection f defined by (7) and (8), the followings are true :*

1) $\deg([\![f]\!]) = \max\{wt_q([\![e_i]\!]) | i = 1, \ldots, d\}$

2) $\deg([\![f^{-1}]\!]) = \max\{wt_q([\![e_i^{-1}]\!]) | i = 1, \ldots, d\}$.

Proof: Using a bijection $e : L \to L$, g can be expressed as

$$g = \psi^{-1} \circ e \circ \psi. \tag{9}$$

From Theorem 7, we get

$$\deg([\![g]\!]) = wt_q([\![e]\!]). \tag{10}$$

Also, from (7) and (9), f can be expressed as

$$f = t \circ \psi^{-1} \circ e \circ \psi \circ s$$

so, from Theorem 7 we have

$$\deg(\llbracket f \rrbracket) = wt_q(\llbracket e \rrbracket). \tag{11}$$

(10) and (11) imply

$$\deg(\llbracket f \rrbracket) = \deg(\llbracket g \rrbracket). \tag{12}$$

Well, from (8) we have

$$\llbracket g \rrbracket = [\llbracket \psi_1^{-1} \circ e_1 \circ \psi_1 \rrbracket, \ldots, \llbracket \psi_d^{-1} \circ e_d \circ \psi_d \rrbracket]$$

and according to the definition of the degree of a tuple of polynomials, we have

$$\deg(\llbracket g \rrbracket) = \max\{deg(\llbracket \psi_i^{-1} \circ e_i \circ \psi_i \rrbracket) | i = 1, \ldots, d\}. \tag{13}$$

Further, from Theorem 7, we get

$$\deg(\llbracket \psi_i^{-1} \circ e_i \circ \psi_i \rrbracket) = wt_q(\llbracket e_i \rrbracket). \tag{14}$$

(12),(13) and (14) imply the first half of the theorem. The second half can be proved in the same way. ♣

IV -4. Univariate Monomials as Grounds

The functions e_i are bijections expressed by univariate polynomials. Polynomials representing bijections are also called *permutation polynomials*, and it is well-known that there are many kinds of such polynomials. However, in this paper, we only deal with those $\llbracket e_i \rrbracket$ which possess the simplest form — the monic monomials. Other forms of $\llbracket e_i \rrbracket$ will be topics for further discussion. We do so for several reasons :

i) It is easy to judge whether a monic monomial represents a bijection or not;

ii) When the bijections e_i are represented by monic monomials $\llbracket e_i \rrbracket$, their inverse functions e_i^{-1} are also represented by monic monomials $\llbracket e_i^{-1} \rrbracket$, so it is easy to compute $\llbracket e_i^{-1} \rrbracket$ from $\llbracket e_i \rrbracket$.

iii) A monic monomial can be readily evaluated.

Now let $[\![e_i]\!], i = 1, \ldots, d$, be a monic monomial in indeterminate u over the finite field L_i of order q^{n_i}, which takes the form of

$$[\![e_i]\!](u) = u^{h_i}, \quad 0 < h_i < q^{n_i} - 1. \tag{15}$$

Since the exponents constitutes a multiplicative semi-group of order $q^{n_i} - 1$, e_i forms a bijection only when h_i and $q^{n_i} - 1$ are relatively prime, i.e., only when $\gcd(h_i, q^{n_i} - 1) = 1$.

Furthermore, suppose that $0 < \overline{h_i} < q^{n_i} - 1$ is the multiplicative inverse element of h_i modulo $(q^{n_i} - 1)$, then $[\![e_i^{-1}]\!]$ forms a monic monomial in indeterminate v :

$$[\![e_i^{-1}]\!](v) = v^{\overline{h_i}}, \quad 0 < \overline{h_i} < q^{n_i} - 1. \tag{16}$$

Since exponential q-weights of $[\![e_i]\!]$ and $[\![e_i^{-1}]\!]$ are equal to the q-weights of h_i and $\overline{h_i}$ respectively, Theorem 8 immediately implies a new theorem:

Theorem 9. *For the bijection defined by (7), (8), (15) and (16), we have*

1) $\deg([\![f]\!]) = \max\{W_q(h_i) | i = 1, \ldots, d\}$

2) $\deg([\![f^{-1}]\!]) = \max\{W_q(\overline{h_i}) | i = 1, \ldots, d\}$.

As mentioned in the beginning of Section IV -3, a small $\deg([\![f]\!])$, but a large $\deg([\![f^{-1}]\!])$ are desirable. Considering Theorem 9, we require that *for all* $i, W_q(h_i)$ *are small, but for some* $i, W_q(\overline{h_i})$ *is large* .

Assume that $\deg([\![f]\!]) = 1$. Now we have $W_q(h_i) = 1$ for all i, and also $W_q(\overline{h_i}) = 1$ for all i. This implies that $\deg([\![f^{-1}]\!]) = 1$, which is not desirable. Hence, it is essential that $\deg([\![f]\!]) \geq 2$. The rest of this chapter will be concerned with the case $\deg([\![f]\!]) = 2$, which can be easily treated. The other cases will also be topics for further discussion.

IV -5. Utilizing Tuples of Quadratic Multivariate Polynomials

For the simplicity of presentation, in this section we only treats the case $d = 1$, and instead of n_i, ψ_i, L_i, e_i and h_i, we will use the notations n, ψ, L, e and h. The results can be easily generalized to the cases $d \geq 2$.

As stated in the end of the last section, here we still assume $\deg([\![f]\!]) = 2$, i.e., $W_q(h) = 2$. The following theorem can be easily obtained.

Theorem 10. *Let p be a prime integer, m, n, q and h be integers satisfying $m > 0, n > 0, q = p^m$, $0 < h < q^n - 1$, and $\gcd(h, q^n - 1) = 1$. Then $p = 2$ is the necessary condition for $W_q(h) = 2$.*

Proof: Assume q be odd. When $W_q(h) = 2$, h can be written as $h = q^j(1 + q^\theta)$, where j and θ are nonnegative integers. Hence h must be even. Also notice that $q^n - 1$ is apparently even. Thus $\gcd(h, q^n - 1)$ must be divided by 2, which contradicts to the assumption of $\gcd(h, q^n - 1) = 1$. Therefore q must be even. Put it in other words, $p = 2$ is the necessary condition for $W_q(h) = 2$. ♣

In the sequel, we will always suppose that $p = 2$, i.e., $q = 2^m$.

Now that $W_q(h) = 2$, as mentioned in the proof of Theorem 10, h can be expressed as

$$h = q^j(1 + q^\theta)$$

where j and θ are nonnegative. Since $\psi^{-1} \circ \langle u^{q^j} \rangle \circ \psi$ is a linear function, we can consider the functions of evaluating the q^jth power together with the affine transformations s and t, between them the function e is inserted. So it suffices to consider the case $j = 0$, and $0 \leq \theta \leq \lfloor n/2 \rfloor$.

If $\theta = 0$, then $h = 2$. In this case, e is a bijection since $\gcd(2, q^n - 1) = 1$. Now consider the n-variate n-tuple representation of the bijection $t \circ \psi^{-1} \circ e \circ \psi \circ s$ over K:

$$[\![t \circ \psi^{-1} \circ e \circ \psi \circ s]\!] = [P_0(x_0, \ldots, x_{n-1}), \ldots, P_{n-1}(x_0, \ldots, x_{n-1})].$$

Since both $p = 2$ and $h = 2$, it is clear that each P_j contains only constant terms and the terms $x_0^2, \ldots, x_{n-1}^2$. In this case, one can quickly solve the following system of quadratic multivariate polynomial equations in indeterminates $x_0, \ldots, x_{n-1}$:

$$\begin{cases} y_0 = P_0(x_0, \ldots, x_{n-1}) \\ \quad \vdots \\ y_{n-1} = P_{n-1}(x_0, \ldots, x_{n-1}). \end{cases}$$

First, taking the system as a system of linear equations in variables $x_0^2, \ldots, x_{n-1}^2$, one can readily solve the new system and get $x_0^2, \ldots, x_{n-1}^2$. Then, one can uniquely determine x_i from x_i^2 (note that $p = 2$). The above algorithm (method) requires about $O(n^3)$ times operations over the field K. So such a system is far from being a good cryptosystem.

Now let us assume that $\theta \neq 0$ furthermore.

From the above discussions, it becomes obvious that we can concentrate our attention upon the case $h = 1+q^\theta, 0 < \theta < \lfloor n/2 \rfloor$. The function $e = \langle u^h \rangle$ is a bijection iff $\gcd(h, q^n - 1) = 1$, which can be restated in another way :

Theorem 11. *Let m, q, θ, n and h be integers satisfying $m > 0, q = 2^m, 0 < \theta < n$ and $h = 1 + q^\theta$. We have $\gcd(h, q^n - 1) = 1$ iff $R_2(\theta) \geq R_2(n)$, where $R_2(\theta)$ (resp. $R_2(n)$) is the 2-rank of θ (resp. n).*

Proof: From Theorem A1 of Appendix , it can be proved that $\gcd(h, q^n - 1) = \gcd(1 + 2^{m\theta}, 2^{mn} - 1) = 1$ is equivalent to $R_2(m\theta) \geq R_2(mn)$. According to the property (R2) of 2-rank functions, we have $R_2(m\theta) = R_2(m) + R_2(\theta)$ and $R_2(mn) = R_2(m) + R_2(n)$, which implies the theorem. ♣

According to Theorem 11, it is necessary that $n \geq 3$. Thus it suffices for us to consider those θ restricted by

$$\theta = b \cdot 2^r, \quad 1 \leq b \leq \ell$$

where r is a nonnegative integer and ℓ is a positive integer such that

$$n = (2\ell + 1) \cdot 2^r, \quad r = R_2(n).$$

In this case, the q-weight of $\overline{h}$ can be calculated from the q-rank of $\overline{h}$, as is stated in the following theorem.

Theorem 12. *For integers m, q, θ, n, h satisfying $m > 0, q = 2^m, 0 \leq \theta < n, h = 1 + q^\theta$, $\gcd(h, q^n - 1) = 1$, the q-weight of the multiplicative inverse element $\overline{h}$ of h modulo $(q^n - 1)$ is given by:*

$$W_q(\overline{h}) = \frac{1}{2}\{(q-1)(n - R_q(\overline{h})) + 1\}.$$

Proof: In Appendix (Lemma A2), we have

$$W_q(q^n - \overline{h}) = W_q(\overline{h}),$$

and also from Appendix (Lemma A3), we have

$$W_q(\overline{h}) + W_q(q^n - \overline{h}) = (q-1)(n - R_q(\overline{h})) + 1,$$

hence,

$$2W_q(\overline{h}) = (q-1)(n - R_q(\overline{h})) + 1$$

and it proves the theorem. ♣

Corollary 3. *Under Theorem 12, we have*

$$\frac{1}{2}q \leq W_q(\overline{h}) \leq \frac{1}{2}\{(q-1)n + 1\}.$$

Proof: $0 \leq R_q(\overline{h}) \leq n-1$, since $0 < \overline{h} < q^n - 1$. Hence $1 \leq n - R_q(\overline{h}) \leq n$, which implies the corollary. ♣

Now we see that, fortunately, $W_q(\overline{h})$ can be increased greatly even when $W_q(h) = 2$. In certain special cases, the q-weight of $\overline{h}$ can be exactly calculated by using the following theorem.

Theorem 13. $R_q(\overline{h}) = 2^r - 1$ *when* $\gcd(b, 2\ell + 1) = 1$.

Proof: $q^{(2\ell+1)\theta} \equiv 1(\mathrm{mod} q^n - 1)$, since $(2\ell+1)\theta = (2\ell+1)2^r b = nb$. Hence

$$2 = 1 + 1 \equiv 1 + q^{(2\ell+1)\theta} \equiv (1 + q^\theta)\sum_{k=0}^{2\ell}(-1)^k q^{\theta k}(\mathrm{mod} q^n - 1).$$

Let $Q = q^{2^r}$, the above equation becomes :

$$\overline{h} \equiv \frac{1}{2}\sum_{k=0}^{2\ell} Q^{bk}(-1)^k(\mathrm{mod} q^n - 1).$$

Since $\gcd(b, 2\ell+1) = 1$, the multiplicative inverse element $\overline{b}$ of b modulo $(2\ell+1)$ exists. Assume that $j = (bk) \bmod (2\ell+1)$, k can be expressed as $k = (\overline{b}j) \bmod (2\ell+1)$. Hence

$$\overline{h} \equiv \frac{1}{2}\sum_{j=0}^{2\ell} Q^j(-1)^{(\overline{b}j)\mathrm{mod}(2\ell+1)}(\mathrm{mod} q^n - 1).$$

Using the relation $1 \equiv q \cdot q^{2^r-1} \cdot Q^{-1}(\mathrm{mod} q^n - 1)$, we get

$$\begin{aligned}\overline{h} &\equiv q^{2^r-1} \cdot (\frac{q}{2}) \cdot \sum_{j=0}^{2\ell} Q^{j-1}(-1)^{(\overline{b}j)\mathrm{mod}(2\ell+1)}(\mathrm{mod} q^n - 1)\\ &\equiv q^{2^r-1} \cdot (\frac{q}{2}) \cdot \sum_{i=0}^{2\ell} Q^{i}(-1)^{[\overline{b}(i+1)]\mathrm{mod}(2\ell+1)}(\mathrm{mod} q^n - 1).\end{aligned}$$

In other words, $\overline{h}$ can be written as

$$\begin{aligned}\overline{h} &\equiv q^{2^r-1} \cdot (\frac{q}{2}) \cdot A(\mathrm{mod} q^n - 1),\\ A &= \sum_{i=0}^{2\ell} Q^{i}(-1)^{[\overline{b}(i+1)]\mathrm{mod}(2\ell+1)}\\ &= 1 + \sum_{i=0}^{2\ell-1} Q^{i}(-1)^{[\overline{b}(i+1)]\mathrm{mod}(2\ell+1)} + Q^{2\ell}.\end{aligned}$$

Apparently, A is not divisible by q. Also, we have

$$0 < A < \sum_{i=0}^{2\ell} Q^i = \frac{Q^{2\ell+1}-1}{Q-1} < Q^{2\ell+1} - 1 = q^n - 1$$

and

$$q^{2^r-1} \cdot (\frac{q}{2}) \cdot A < \frac{1}{2}Q\frac{Q^{2\ell+1}-1}{Q-1} = \frac{Q^{2\ell+1}-1}{2(1-1/Q)} < Q^{2\ell+1} - 1 = q^n - 1.$$

Therefore, from $0 < \overline{h} < q^n - 1$, we have

$$\overline{h} = q^{2^r-1} \cdot (\frac{q}{2}) \cdot A, \quad q \text{ does not divide } A$$

(Notice : not $\equiv$, but $=$). Hence $R_q(\overline{h}) = 2^r - 1$. ♣

From Theorem 12 and Theorem 13, it can be shown that, when n is an odd integer ≥ 3 $(r = 0)$ and θ is relatively prime to n, the q-rank of $\overline{h}$ becomes zero, and the q-weight of $\overline{h}$ reaches its maximum — $\frac{1}{2}\{(q-1)n+1\}$. Thus, when $R_q(\overline{h}) = 0$, we get

$$\tau_{up}([\![f^{-1}]\!]) > \{(\frac{\varepsilon}{2})(2^m - 1)\}^n \cdot (\frac{n}{2\pi})^{1/2}$$

where ε is the base of natural logarithms. The above inequality tells us that the n-variate polynomial n-tuple representation of the function $f^{-1} = s^{-1} \circ \psi^{-1} \circ e^{-1} \circ \psi \circ t^{-1}$, contains approximately exponentially in m and n many number of nonzero terms, and writing down all those terms is practically impossible. The correctness of the inequality can be ascertained by a simple calculation using the definition of τ_{up}, Theorem 9, Theorem 12, and the Stirling's formula on factorials.

IV -6. Proof of Theorem 1 and Theorem 3

In Sections IV -3, -4 , -5, we discussed in detail specializations of C_0^* . The resulting asymmetric cryptosystem is nothing but our C^* defined in Definition 1. Therefore we can see that Theorem 1 really holds. And the first half of Theorem 3 follows from Theorem 9 and Corollary 3 and from that $n_1 \leq \cdots \leq n_d$. The second half of Theorem 3 immediately follows from the discussions made in the end of Section IV-5.

V . CONCLUDING REMARKS

On a basis different from the previous, this paper has proposed and analyzed an asymmetric cryptosystem C^* which can serve for both digital signatures and encryption.

An advantage of C^* over the previous asymmetric cryptosystems is that both secret and public transfromations can be done in complexity much less than $O(N^3)$ for a message block of size N. Actually, we have implemented C^* with the languages "C" and *occam* on 32-bit microprocessors and verified high performance of C^* .

The description length of a key for C^* is greater than that of previous systems with the same block size. However, this is not always a demerit as mentioned in Section III -5.

Thus the present authors believe that C^* is a cryptosystem worth investigating for everybody interested in high-speed cryptographic communications.

ACKNOWLEDGMENT

The authors wish to thank Youichi Takashima for his help on making numerical examples of C^* and Yuliang Zheng for his kind interpretation of the Chinese papers [7, 8]. This work was supported in part by the Ministry of Educations, Science and Culture under Grant-in-Aid for Encouragement of Young Scientists # 62750283.

REFERENCES

[1] Diffie,W. and Hellman,M.E., "New directions in cryptography," IEEE Transactions on Information Theorey, IT-22, 6, pp.644-654, (Nov. 1976).

[2] Cardoza,E., Lipton,R. and Meyer,A.R.,"Exponential space complete problems for Petri nets and commutative semigroups," Conf. Record of the 8th Annual ACM Symposium on Theory of Computing, pp.50-54, (1976).

[3] Garey,M.R. and Johnson,D.S., Computer and Intractability: A guide to the theory of NP-comptleteness, Freeman,(1979).

[4] Matsumoto,T., Imai,H., Harashima,H. and Miyakawa,H., "A class of asymmetric cryptosystems using obscure representations of enciphering functions," 1983 National Convention Record on Information Systems, IECE Japan, S8-5, (Sept. 1983) (in Japanese).

[5] Matsumoto,T., Harashima,H. and Imai,H., "A theory of constructing multivariate-polynomial-tuple asymmetric cryptosystems," Proceedings of 1986 Symposium on Cryptography and Information Security, E2, Susono, Japan, (Feb. 1986) (in Japanese).

[6] Fell,H. and Diffie,W., "Analysis of a public key approach based on polynomial substitution," Advances in Cryptology — CRYPTO '85, Springer, pp.340-349, (1986).

[7] Zhou,T., "Boolean public key cryptosystem of the second order," Journal of China Institute of Communications, Vol.5, No.3, pp.30-37, (July 1984) (in Chinese).

[8] Zhou,T., "A note on boolean public key cryptosystem of the second order," Journal of China Institute of Communications, Vol.7, No.1, pp.85-92, (Jan. 1986) (in Chinese).

[9] Lidle,R. and Niederreiter,H., *Finite Fields*, Addison-Wesley (1983).

[10] Rivest,R.L., Shamir,A. and Adleman,L., "A mehtod of obtaing digital signatures and public key cryptosystems," Communications of ACM, Vol.21, No.2, pp.120-126, (Feb.1978).

[11] Takahashi,I., "Switching functions constructed by Galois extension fields," Information and Control, Vol.48, pp.95-108, (1983).

[12] Matsumoto,T.,Imai,H.,Harashima,H. and Miyakawa,H., "A cryptographically useful theorem on the connection between uni and multivariate polynomials," Transactions of the Institute of Electronics and Communication Engineers, Vol.E68, No.3, pp.139-146, (March 1985).

APPENDIX

Lemma A1. *If integers a, b, c and f satisfy $a > b > c \geq 0$ and $a = bf + c$, then*

$$\gcd(2^a \pm 1, 2^b + 1) = \gcd(2^b + 1, 2^c \pm (-1)^f),$$
$$\gcd(2^a + 1, 2^b - 1) = \gcd(2^b - 1, 2^c + 1).$$

Proof: From

$$2^a \pm 1 = 2^{bf} 2^c \pm 1$$

and

$$\begin{aligned} 2^{bf} &= \{\mp 1 + (2^b \pm 1)\}^f \\ &= \sum_{j=0}^{f} \binom{f}{j} (\mp 1)^j (2^b \pm 1)^{f-j} \\ &= (2^b \pm 1)\{\sum_{j=0}^{f-1} \binom{f}{j} (\mp 1)^j (2^b \pm 1)^{f-j-1}\} + (\mp 1)^f, \end{aligned}$$

we get

$$2^a \pm 1 = (2^b + 1)\{\sum_{j=0}^{f-1} \binom{f}{j} (-1)^j (2^b + 1)^{f-j-1}\} 2^c + (-1)^f 2^c \pm 1,$$

$$2^a + 1 = (2^b - 1)\{\sum_{j=0}^{f-1} \binom{f}{j} (2^b - 1)^{f-j-1}\} 2^c + 2^c + 1.$$

♣

Theorem A1. *If integers a, b, d satisfy $a > 0, b > 0, d = \gcd(a, b)$, then*

$$\gcd(2^a + 1, 2^b - 1) = \begin{cases} 1; & R_2(a) \geq R_2(b) \\ 2^d + 1; & R_2(a) < R_2(b). \end{cases}$$

Proof: By applying Lemma A1 iteratively, we can find that $\gcd(2^a + 1, 2^b - 1)$ is equivalent to $\gcd(2^d \pm 1, 2^0 + 1) = \gcd(2^d \pm 1, 2) = 1$ or $\gcd(2^d \pm 1, 2^0 - 1) = \gcd(2^d + 1, 0) = 2^d + 1$. Now from the definition of R_2 and Lemma A1, we have

$$\begin{aligned} R_2(a) < R_2(b) &\Longleftrightarrow R_2(a) = R_2(d) < R_2(b) \\ &\Longleftrightarrow \begin{cases} R_2(a/d) = R_2(a) - R_2(d) = 0 \\ R_2(b/d) = R_2(b) - R_2(d) > 0 \end{cases} \\ &\Longleftrightarrow (-1)^{a/d} = -1 \text{ and } (-1)^{b/d} = 1 \\ &\Longleftrightarrow \begin{cases} \gcd(2^a + 1, 2^d + 1) = \gcd(2^d + 1, 2^0 + (-1)) \\ \qquad\qquad\qquad\qquad\quad = 2^d + 1 \\ \gcd(2^b - 1, 2^d + 1) = \gcd(2^d + 1, 2^0 - 1) \\ \qquad\qquad\qquad\qquad\quad = 2^d + 1 \end{cases} \\ &\Longleftrightarrow (2^d + 1) | \gcd(2^a + 1, 2^b - 1) \end{aligned}$$

which proves the theorem. ♣

Lemma A2. *For integers m, q, θ, n and h with $m > 0, q = 2^m, 0 \leq \theta < n, h = 1 + q^\theta, \gcd(h, q^n - 1) = 1$, the multiplicative inverse element $\overline{h}$ of h satisfies*

$$W_q(q^n - \overline{h}) = W_q(\overline{h}). \qquad (A2 - 1)$$

Proof: Let the q-ary representaiton of $\overline{h}$ be $\overline{h} = \sum_{i=0}^{n-1} q^i \xi_i, (0 \leq \xi_i < q)$. By introducing an integer k, $(1 + q^\theta) \cdot \overline{h}$ can be writen as $k(q^n - 1) + 1$. Hence,

$$q^n - \overline{h} = q^\theta \overline{h} - (k-1)(q^n - 1) \qquad (A2-2)$$

Because

$$\begin{aligned} q^\theta \overline{h} &= q^n \sum_{j=0}^{\theta-1} q^j \xi_{n-\theta+j} + \sum_{j=\theta}^{n-1} q^j \xi_{j-\theta} \\ &= \sum_{j=\theta}^{n-1} q^j \xi_{(j-\theta) \bmod n} + (\sum_{j=0}^{\theta-1} q^j \xi_{n+j-\theta})(q^n - 1), \end{aligned}$$

we get

$$q^\theta - \overline{h} = \sum_{j=0}^{n-1} q^j \xi_{(j-\theta) \bmod n} + (\sum_{j=0}^{\theta-1} q^j \xi_{n+j-\theta} - (k-1))(q^n - 1)$$

from (A2-2). Also, $q^n - \overline{h} < q^n - 1$ since $\overline{h} > 1$. Hence

$$q^\theta - \overline{h} = \sum_{j=0}^{n-1} q^j \xi_{(j-\theta) \bmod n}$$

which implies (A2-1). ♣

Lemma A3. *If an integer a satisfies $1 \leq a \leq q^n - 1$, then*

$$W_q(a) + W_q(q^n - a) = (q-1)(n - \lambda) + 1,$$

where $\lambda = R_q(a)$.

Proof: We can uniquely determine a positive integer b such that $a = q^\lambda \cdot b$ and b is not divisible by q. Thus

$$W_q(a) = W_q(b). \qquad (A3-1)$$

Also, from $q^n - a = q^n - q^\lambda b = q^\lambda(q^{n-\lambda} - b)$, we get

$$W_q(q^n - a) = W_q(q^{n-\lambda} - b). \qquad (A3-2)$$

Since b is not divisible by q, and can be expressed as

$$b = q\nu + \mu + 1, \quad (0 \leq \mu < q - 1),$$

by using the properties (W1) and (W2) of W_q, we get

$$\begin{aligned} W_q(b) &= W_q(q\nu + (\mu + 1)) \\ &= W_q(q\nu) + W_q(\mu + 1) \\ &= W_q(q\nu) + (W_q(\mu) + 1) \\ &= (W_q(q\nu) + W_q(\mu)) + 1 \\ &= W_q(q\nu + \mu) + 1 \\ &= W_q(b - 1) + 1, \end{aligned}$$

i.e.,

$$W_q(b) = W_q(b - 1) + 1. \qquad (A3 - 3)$$

Furthermore, from $(b-1)+(q^{n-\lambda}-b) = q^{n-\lambda}-1$ and the property (W3) of W_q, we get

$$W_q(b - 1) + W_q(q^{n-\lambda} - b) = (n - \lambda)(q - 1). \qquad (A3 - 4)$$

Thus, by $(A3-1)$, $(A3-2)$, $(A3-3)$, and $(A3-4)$, we have the following:

$$\begin{aligned} W_q(a) + W_q(q^n - a) &= W_q(b) + W_q(q^{n-\lambda} - b) \\ &= 1 + W_q(b - 1) + W_q(q^{n-\lambda} - b) \\ &= 1 + (n - \lambda)(q - 1) \ \clubsuit. \end{aligned}$$