Two Varieties of Finite Automaton Public Key Cryptosystem and Digital Signatures

Tao Renji(陶仁骥) and Chen Shihua(陈世华)
(*Institute of Software, Academia Sinica*)

Received May 17, 1985.

Abstract

This paper gives two varieties of the public key cryptosystem in [1] which can also be used to implement digital signatures.

1. Introduction

The concept of public key cryptosystems was introduced by Diffie and Hellman in 1976 [2]. In a public key cryptosystem, each user has a public encryption algorithm E and a secret decryption algorithm D. These algorithms satisfy the following conditions. 1) D is an inverse of E. 2) E and D are easy to calculate. 3) Each easily calculated algorithm equivalent to D is computationally infeasible to derive from E. And the public key cryptosystem can be used to implement digital signatures if E is an inverse of D. Many concrete schemes of public key cryptosystem have been invented [3-6, 9-15]. Among the others, the *RSA* cryptosystem is drawn from number theory which can be use to implement digital signatures, and the trapdoor knapsack system from combinatorial mathematics [3,9]. All the systems [3-6, 9-15] are block cryptosystems. In 1984, we introduced a public key cryptosystem based on invertibility theory of finite automata[1] of which security rests on the difficulties of finding weak inverses of nonlinear finite automata and of factoring matrix polynomials over Galois field. Because this is, to our knowledge, the first sequential (or stream) public key cryptosystem, its implementation is easy and the size of its public key is relatively small. This paper gives two varieties of the public key cryptosystem in [1] which can also be used to implement digital signatures and have slight improvement in the size of public key.

2. Preliminaries

For any finite automata $M = \langle X, Y, S, \delta, \lambda \rangle$ and $M' = \langle Y, X, S', \delta', \lambda' \rangle$, M' is said to be a weak inverse with delay τ of M if for any s in S there exists s' in S' such that s' and s is a match pair with delay τ, i.e. for any $x_0, x_1, \cdots$ in X, the equation

$$\lambda'(s', \lambda(s, x_0x_1\cdots)) = x_{-\tau}\cdots x_{-1}x_0x_1\cdots$$

holds for some $x_{-\tau}, \cdots, x_{-1}$ in X. And M' is said to be an inverse with delay τ of M if for any s in S and s' in S', s' and s is a match pair with delay τ.

Proposition 1. *Let* $M = \langle X, Y, S, \delta, \lambda \rangle$ *and* $M' = \langle Y, X, S', \delta', \lambda' \rangle$ *be two finite automata and* $X = Y$. *Then for any states* s *of* M *and* s' *of* M', s' *and* s *is a match pair with delay free if and only if* s *and* s' *is a match pair with delay free.*

Proof. Suppose that s' and s is a match pair with delay free. We prove by reduction to absurdity that s and s' is a match pair with delay free. Suppose to the contrary that for some sequence $y(0)\,y(1)\cdots$ over Y, $\lambda(s, \lambda'(s', y(0)\,y(1)\cdots))$, denoted by $y'(0)\,y'(1)\cdots$, is not equal to $y(0)\,y(1)\cdots$. Then $y(0)\cdots y(n) \neq y'(0)\cdots y'(n)$ for some $n \geqslant 0$. Since s' and s is a patch pair with delay free, we have $\lambda'(s', y'(0)\cdots y'(n)) = \lambda'(s', y(0)\cdots y(n))$. From $X = Y$, $\lambda'(s', Y^{n+1}) \neq X^{n+1}$. It follows that there exists $x''(0)\cdots x''(n)$ in $X^{n+1} - \lambda'(s', Y^{n+1})$. Denote $\lambda(s, x''(0)\cdots x''(n)) = y''(0)\cdots y''(n)$. Since s' and s is a match pair with delay free, we have $\lambda'(s', y''(0)\cdots y''(n)) = x''(0)\cdots x''(n)$. Thus $x''(0)\cdots x''(n)$ is in $\lambda'(s', Y^{n+1})$. This is a contradiction. From symmetry, the proposition is proven.

Let $M_0 = \langle X, Y', S_0, \delta_0, \lambda_0 \rangle$ and $M_1 = \langle Y', Y, S_1, \delta_1, \lambda_1 \rangle$ be two finite automata. We use $C(M_0, M_1)$ to denote the finite automaton $M = \langle X, Y, S_0 \times S_1, \delta, \lambda \rangle$, where

$$\delta(\langle s_0, s_1 \rangle, x) = \langle \delta_0(s_0, x),\ \delta_1(s_1, \lambda_0(s_0, x)) \rangle,$$

$$\lambda(\langle s_0, s_1 \rangle, x) = \lambda_1(s_1, \lambda_0(s_0, x)).$$

Let $M_1' = \langle Y, Y', S_1, \delta_1', \lambda_1' \rangle$ be a t-order input-memory finite automaton, defined by [16,p. 10]

$$y'(i) = f(y(i-t), \cdots, y(i)), \qquad i = 0,1,\cdots. \tag{1}$$

Let $M_0' = \langle Y', X, S_0', \delta_0', \lambda_0' \rangle$ be a r-order input-memory finite automaton defined by [16, p. 10]

$$x'(i) = g(y'(i-r), \cdots, y'(i)), \qquad i = 0,1,\cdots. \tag{2}$$

We use $C'(M_1', M_0')$ to denote the $(t+r)$-order input-memory finite automaton with input alphabet Y and output alphabet X, defined by

$$x'(i) = g(f(y(i-r-t), \cdots, y(i-r)), \cdots\cdots, f(y(i-t), \cdots, y(i))), \qquad i = 0,1,\cdots. \tag{3}$$

Theorem 1. *Assume that* M_1' *is a weak inverse with delay free of* M_1. *Assume that* M_0 *is a* (τ, r)*-order memory finite automaton and for any states* $s_0 = \langle y'(-r), \cdots, y'(-1), x(-\tau), \cdots, x(-1) \rangle$ *of* M_0 *and* $s_0' = \langle y'(-r), \cdots, y'(-1) \rangle$ *of* M_0', s_0' *and* s_0 *is a match pair with delay* τ. *Then for any state* s_1 *of* M_1 *there exist* $y(-t), \cdots, y(-1)$ *in* Y *such that for any* $y(-r-t), \cdots, y(-t-1)$ *in* Y *and any* $x(-\tau), \cdots, x(-1)$ *in* X *the state* $\langle y(-r-t), \cdots, y(-1) \rangle$ *of* $C'(M_1', M_0')$ *and the state* $\langle s_0, s_1 \rangle$ *of* $C(M_0, M_1)$ *is a match pair with delay* τ, *where* $s_0 = \langle y'(-r), \cdots, y'(-1), x(-\tau), \cdots, x(-1) \rangle$ *is a state of* M_0 *and*

$$y'(i) = f(y(i-t), \cdots, y(i)), \qquad i = -1, \cdots, -r. \tag{4}$$

Proof. Given any state s_1 in M_1 and $y(-r-t), \cdots, y(-t-1)$ in Y. Since M_1' is a weak inverse with delay free of M_1, there exists a state $s_1' = \langle y(-t), \cdots, y(-1) \rangle$ of M_1' such that s_1' and s_1 is a match pair with delay free. For any $x(0), x(1), \cdots$ in X, we denote

$$\lambda_0(s_0, x(0)x(1)\cdots) = y'(0)y'(1)\cdots,$$

$$\lambda_1(s_1, y'(0)y'(1)\cdots) = y(0)y(1)\cdots.$$

Thus $\lambda_1'(s_1', y(0)y(1)\cdots) = y'(0)y'(1)\cdots$ holds. It follows that

$$y'(i) = f(y(i-t), \cdots, y(i)), \qquad i = 0,1, \cdots. \tag{5}$$

From the hypothesis of the Theorem, s_0' and s_0 is a match pair with delay τ, where $s_0' = \langle y'(-r), \cdots, y'(-1)\rangle$. Then $\lambda_0'(s_0', y'(0)y'(1)\cdots) = x'(0)\cdots x'(\tau-1)x(0)x(1)\cdots$ holds for some $x'(0), \cdots, x'(\tau-1)$ in X.It follows immediately that

$$x(i-\tau) = g(y'(i-r), \cdots, y'(i)), \qquad i = \tau, \tau+1, \cdots. \tag{6}$$

From (4), (5) and (6), we have

$$x(i-\tau) = g(f(y(i-r-t), \cdots, y(i-r)), \cdots\cdots, f(y(i-t), \cdots, y(i))),$$

$$i = \tau, \tau+1, \cdots.$$

Thus $\langle y(-r-t), \cdots, y(-1)\rangle$ and $\langle s_0, s_1\rangle$ is a match pair with delay τ.

Theorem 2. *Assume that M_0' is an inverse with delay r of M_0 and M_1' is a weak inverse with delay τ of M_1, where M_0' and M_1' are defined by (2) and (1) respectively. Then $C'(M_1', M_0')$ is a weak inverse with delay $r+\tau$ of $C(M_0, M_1)$.*

Proof. Given any state $\langle s_0, s_1\rangle$ of $C(M_0, M_1)$. Since M_1' is a weak inverse with delay τ of M_1, there exists a state $s_1' = \langle y(-t), \cdots, y(-1)\rangle$ of M_1' such that s_1' and s_1 is a match pair with delay τ. Let $s' = \langle y(-r-t), \cdots, y(-1)\rangle$ be a state of $C'(M_1', M_0')$,where $y(-r-t), \cdots, y(-t-1)$ are arbitrary elements in Y. Below we prove that s' and $\langle s_0, s_1\rangle$ are a match pair with delay $r+\tau$. Given any $x(0), x(1), \cdots$ in X, denote

$$\lambda_0(s_0, x(0)x(1)\cdots) = y'(0)y'(1)\cdots,$$

$$\lambda_1(s_1, y'(0)y'(1)\cdots) = y(0)y(1)\cdots.$$

Let $\lambda_1'(s_1', y(0)y(1)\cdots) = y''(0)y''(1)\cdots$. Then $y''(i) = y'(i-\tau)$, $i = \tau, \tau+1, \cdots$. It follows that

$$y'(i-\tau) = f(y(i-t), \cdots, y(i)), \qquad i = \tau, \tau+1, \cdots. \tag{7}$$

Let $\lambda'(s', y(0)y(1)\cdots) = x'(0)x'(1)\cdots$, λ' being the output function of $C'(M_1', M_0')$. It is evident that

$$x'(i) = g(f(y(i-r-t), \cdots, y(i-r)), \cdots\cdots, f(y(i-t), \cdots, y(i))), i = 0,1, \cdots.$$

Using (7), we have

$$x'(i) = g(y'(i-r-\tau), \cdots, y'(i-\tau)), \qquad i = r+\tau, r+\tau+1, \cdots. \tag{8}$$

Since M_0' is an inverse with delay r of M_0, for any state s_0' of M_0',

$$\lambda_0'(s_0', y'(0)y'(1)\cdots) = x(-r)\cdots x(-1)x(0)x(1)\cdots$$

holds for some $x(-r), \cdots, x(-1)$ in X. It follows that

$$x(i-r) = g(y'(i-r), \cdots, y'(i)), \qquad i = r, r+1, \cdots. \tag{9}$$

From (8) and (9), we have $x'(i) = x(i - r - \tau)$, $i = r + \tau$, $r + \tau + 1, \cdots$.

The following Proposition is a special case of the Corollary in [1].

Proposition 2. *Let* $s' = \langle y(-r-t), \cdots, y(-1)\rangle$ *be a state of* $C'(M_1', M_0')$, $s_1' = \langle y(-t), \cdots, y(-1)\rangle$ *and* $s_0' = \langle y'(-r), \cdots, y'(-1)\rangle$, *where* $y'(i) = f(y(i-t), \cdots, y(i))$, $i = -1, \cdots, -r$. *Then* $\langle s_1', s_0'\rangle$ *is a state of* $C'(M_1', M_0')$ *and equivalent to* s'.

Below we give an expression for (3). It is well known that any n-ary function over $GF(q)$ can be expressed by a polynomial

$$\sum_{i_1, \cdots, i_n = 0}^{q-1} b_{i_1 \cdots i_n} x^{i_1} \cdots x^{i_n}$$

with coefficients $b_{i_1 \cdots i_n}$ s in $GF(q)$. Let X, Y and Y' be column vector spaces over $GF(q)$ with dimension l, m and m', respectively. We use $\pi(u_0, \cdots, u_j)$ to denote a column vector of which all components are just monomials of some components of $u_0, \cdots, u_j$ containing at least a component of u_0 and of u_j. $\pi(u_0, \cdots, u_j)$ is said to be the monomial vector with span $j+1$. Clearly, f can be expressed as the following

$$f(y_{-t}, \cdots, y_0) = F + \sum_{k=0}^{t} \sum_{h=k}^{t} F_{kh} \pi(y_{-h}, \cdots, y_{-k}), \tag{10}$$

where F is a m'-dimensional column vector over $GF(q)$ and F_{hk} is a $m' \times n$ matrix over $GF(q)$ for some n.

Let M_0' be linear, that is,

$$g(y'_{-r}, \cdots, y'_{-1}, y'_0) = \sum_{j=0}^{r} B_j y'_{-j} \tag{11}$$

for some $l \times m'$ matrices $B_0, \cdots, B_r$ over $GF(q)$. Then (3) can be expressed by

$$x'(i) = \sum_{j=0}^{r} B_j [F + \sum_{k=0}^{t} \sum_{h=k}^{t} F_{kh} \pi(y(i-j-h), \cdots, y(i-j-k))], \qquad i = 0,1, \cdots,$$

that is,

$$x'(i) = C + \sum_{j=0}^{r+t} \sum_{k=j}^{r+t} C_{jk} \pi(y(i-k), \cdots, y(i-j)), \qquad i = 0,1, \cdots, \tag{12}$$

where

$$C = (\sum_{j=0}^{r} B_j) F,$$

$$C_h(z) = B(z) F_h(z), \qquad h = 0,1, \cdots, t,$$

$$B(z) = \sum_{j=0}^{r} B_j z^j,$$

(13)

$$F_h(z)=\sum_{j=0}^{t-h}F_{j,j+h}z^j,\qquad h=0,1,\cdots,r,$$

$$C_h(z)=\sum_{j=0}^{r+t-h}C_{j,j+h}z^j,\qquad h=0,1,\cdots,r.$$

3. Several Public Key Cryptosystems and Digital Signatures

Throughout this section, X and Y are taken as the column vector spaces over $GF(q)$ with dimension l and m, respectively.

To construct a public key cryptosystem based on the invertibility theory of finite automata, people can choose a common q and l, and take $m=l$ for the sake of digital signatures. In other words, the cleantext alphabet and ciphertext alphabet are all the same to every users, that is, l-dimensional column vector space over $GF(q)$.We first restate the public key cryptosystem based on the invertibility theory of finite automata introduced by the authors in [1], then introduce two varieties.

3.1. *The system in* [1]

An user, say A, can choose his (her) own encryption key and decryption key according to the following procedure. 1) Choose a τ-order input-memory linear finite automaton $M'_1=\langle Y, X, S'_1, \delta'_1, \lambda'_1\rangle$, M'_1 being an inverse with delay τ, defined by

$$x''(i)=\sum_{j=0}^{\tau}A'_jy(i-j),\qquad i=0,1,\cdots \tag{14}$$

(for detail see [16] §2.6). 2) From M'_1 make a (τ,τ)-order memory linear finite automaton $M_1=\langle X,Y,S_1,\delta_1,\lambda_1\rangle$,$M'_1$ being an inverse with delay τ of M_1, defined by

$$y(i)=\sum_{j=1}^{\tau}A_jy(i-j)+\sum_{j=0}^{\tau}B_jx'(i-j),\qquad i=0,1,\cdots$$

(for detail see [16] §2.6). 3) Choose a $(r+1)$-ary nonlinear function $f(v_0,\cdots,v_r)$ over X such that for any $v_1,\cdots,v_r$ in X, $f(v_0,\cdots,v_r)$ as an unary function of argument v_0 is invertible. 4) From M_1 and f make a $(\tau+r,\ \tau)$ -order memory finite automaton M defined by

$$y(i)=\sum_{j=1}^{\tau}A_jy(i-j)+\sum_{j=0}^{\tau}B_jf(x(i-j),\cdots,x(i-j-r)),\qquad i=0,1,\cdots. \tag{15}$$

5) Choose arbitrary $x(-1),\cdots,x(-r)$ in X. Then $(M,\ s)$ is the public encryption key of user A, where $s=\langle x(-r),\ \cdots,\ x(-1)\rangle$. 6) From f make a $(r+1)$-ary function f' such that $f'(f(v_0,\cdots,v_r),v_1,\cdots,v_r)=v_0$ holds for any $v_0,\cdots,v_r$ in X. Then (M'_1,f') is the secret decryption key of user A.

When another user B wishes to send a message $x(0)\cdots x(n)$ to user A in secrecy, B first extends arbitrary τ digits $x(n+1),\ \cdots,\ x(n+\tau)$ in X, then chooses arbitrary $y(-1),\ \cdots,\ y(-\tau)$ in Y and $x(-r-1),\ \cdots,\ x(-r-\tau)$ in X and calculates, using A's public key, the ciphertext $y(0)\cdots y(n+\tau)$ according to (15) which is sent to user A thereafter. On receipt, user A first calculates values $x''(\tau),\cdots,x''(n+\tau)$ according to (14), then calculates values $x(0),\cdots,x(n)$, using f' in A's secret key and s in A's public key, by

$$x(i)=f'(x''(i+\tau),x(i-1),\cdots,x(i-r)),\qquad i=0,1,\cdots,n.$$

This public key cryptosystem can be slightly modified to implement digital signatures. That is, A's public key is extended to (M, s, s_0), where $s_0=\langle y(-\tau),\cdots, y(-1), x(-\tau-r),\cdots, x(-1)\rangle$ satisfying

$$\sum_{j=1}^{\tau-i}A_{i+j}y(-j)+\sum_{j=1}^{\tau-i}B_{i+j}f(x(-j),\cdots,x(-j-r))=0,$$

$$i=0,\cdots,\tau-1.$$

Notice that, in case of $f(0,\cdots,0)=0$, we may take $s_0=\langle 0,\cdots,0\rangle$. When user A needs to sign a message $y(0)\cdots y(n)$, A first extends arbitrary τ digits $y(n+1),\cdots,y(n+\tau)$ in Y, then calculates $x(0)\cdots x(n+\tau)$ by

$$x(i)=f'(x'(i),x(i-1),\cdots,x(i-r)),\qquad i=0,1,\cdots,n+\tau,$$

using A's secret key and s in A's public key, where $x'(0)\cdots x'(n+\tau)=\lambda_1'(0,y(0)\cdots y(n+\tau))$. Every one, say B, validates A's signature $x(0)\cdots x(n+\tau)$ on message $y(0)\cdots y(n)$ by calculating $\lambda(\delta(s_0,x(0)\cdots x(\tau-1)),x(\tau)\cdots x(n+\tau))$ using A's public key, which is equal to $y(0)\cdots y(n)$.

3.2. *The first variety*

Symmetrically, using Theory 1, we can construct a public key cryptosystem which can also be used to implement digital signatures.

A user, say A, can design his (her) own secret key and public key as follows. 1) Choose a r-order input-memory linear finite automaton $M_0'=\langle Y, X, S_0', \delta_0', \lambda_0'\rangle$, M_0' being a weak inverse with delay τ, defined by (2) (for detail see [16] §2.4). 2) From M_0', make a (τ, r)-order memory linear finite automaton $M_0=\langle X, Y, S_0, \delta_0, \lambda_0\rangle$ such that M_0' is a weak inverse with delay τ of M_0 and for any states of the form $s_0=\langle y'(-r),\cdots,y'(-1),x(-\tau),\cdots,x(-1)\rangle$ of M_0 and $s_0'=\langle y'(-r),\cdots,y'(-1)\rangle$ of M_0', s_0' and s_0 is a match pair with delay τ (for detail see [16] p. 141 Theorem 8). 3) Choose a t-order input-memory nonlinear finite automaton $M_1'=\langle Y, Y, S_1', \delta_1', \lambda_1'\rangle$ defined by (1) and a finite automaton $M_1=\langle Y, Y, S_1, \delta_1, \lambda_1\rangle$ such that M_1' is a weak inverse with delay free of M_1. 4) From M_0' and M_1', make the finite automaton $C'(M_1', M_0')$ expressed in the form of (12). 5) Choose a state s_1 of M_1 and a state $s_1'=\langle y(-t),\cdots,y(-1)\rangle$ of M_1' such that s_1' and s_1 is a match pair with delay free. Choose $y(-r-t),\cdots,y(-t-1)$ in Y. Let $s_0'=\langle y'(-r),\cdots,y'(-1)\rangle$, where $y'(i)=f(y(i-t),\cdots,y(i-1))$, $i=-1,\cdots,-r$. Then $(C'(M_1', M_0'), \langle y(-r-t),\cdots,y(i-1)\rangle,\tau)$ and $\langle M_0, M_0', M_1, s_0', s_1\rangle$ are the public key and the secret key of user A, respectively.

When another user B wishes to send a message $y(0)\cdots y(n)$ to user A in secrecy, B first extends arbitrary τ digits $y(n+1),\cdots,y(n+\tau)$ in Y, then calculates $x'(0)\cdots x'(n+\tau)$ according to (12) using A's public key. The ciphertext $x'(0)\cdots x'(n+\tau)$ is sent to A. On receipt, user A first calculates $\lambda_0(0, x'(0)\cdots x'(n+\tau)-\lambda_0'(s_0', 0^{n+\tau+1}))$ denoted by $y''(0)\cdots y''(n+\tau)$ using M_0, M_0' and s_0' in A's secret key, then calculates $\lambda_1(s_1, y''(\tau)\cdots y''(n+\tau))$ using M_1 and s_1 in A's secret key which is equal to $y(0)\cdots y(n)$.

To prove $\lambda_1(s_1, y''(\tau)\cdots y''(n+\tau))=y(0)\cdots y(n)$, denote $\lambda_1'(s_1', y(0)\cdots y(n+\tau))=y'(0)\cdots y'(n+\tau)$. From Proposition 2, we have $\lambda_0'(s_0', y'(0)\cdots y'(n+\tau))=x'(0)\cdots x'(n+\tau)$. Since M_0' is linear, we have $x'(0)\cdots x'(n+\tau)-\lambda_0'(s_0', 0^{n+\tau+1})=\lambda_0'(0, y'(0)\cdots y'(n+\tau))$. Since M_0' is a weak inverse with delay τ of M_0 and they are linear, we have $y''(0)\cdots y''(n+\tau)=\lambda_0(0,$

$\lambda'_0(0, y'(0)\cdots y'(n+\tau)) = 0^{\tau} y'(0)\cdots y'(n)$. It follows immediately that $y''(\tau)\cdots y''(n+\tau) = y'(0)\cdots y'(n)$. Since s'_1 and s_1 is a match pair with delay free, from Proposition 1, s_1 and s'_1 is a match pair with delay free also. Therefore,

$$\lambda_1(s_1,\ y''(\tau)\ \cdots\ y''(n+\tau)) = \lambda_1(s_1,\ \lambda'_1(s'_1, y(0)\cdots y(n))) = y(0)\cdots y(n).$$

This public key system can be used to implement digital signatures. User A can sign a message $x(0)\cdots x(n)$ by the following steps. 1) User A first chooses arbitrary 2τ digits $x(-\tau), \cdots, x(-1), x(n+1), \cdots, x(n+\tau)$ in X. 2) Then A calculates $y(0)\cdots y(n+\tau) = \lambda_1(s_1, \lambda_0(s_0, x(0)\cdots x(n+\tau))$ using M_0, M_1, s'_0 and s_1 in A's secret key, s_0 being $\langle y'(-r), \cdots, y'(-1), x(-\tau), \cdots, x(-1)\rangle$. Every one, say B, can validate A's signature $y(0)\cdots y(n+\tau)$ on $x(0)\cdots x(n)$ by calculating $\lambda'(\langle y(-r-t), \cdots, y(-1)\rangle, y(0)\cdots y(n+\tau))$ using A's public key, where λ' is the output function of $C'(M'_1, M'_0)$. From Theorem 1, it is easy to see that $\lambda'(\langle y(-r-t), \cdots, y(-1)\rangle, y(0)\cdots y(n+\tau))$ is equal to $x'(0)\cdots x'(\tau-1)\,x(0)\cdots x(n)$ for some $x'(0), \cdots, x'(\tau-1)$ in X.

3.3. *The second variety*

Another public key cryptosystem which can also be used to implement digital signatures is based on Theorem 2. A user, say A, can design his (her) own secret key and public key as follows. 1) Choose a r-order input-memory linear finite automaton $M'_0 = \langle Y, X, S'_0, \delta'_0, \lambda'_0\rangle$, M'_0 being an inverse with delay r, defined by (2) (for detail see [16] §2.6). 2) From M'_0, make a (r, r)-order memory linear finite automaton $M_0 = \langle X, Y, S_0, \delta_0, \lambda_0\rangle$ such that M'_0 is an inverse with delay r of M_0 (for detail see [16] p.169 Theorem 3). 3) Choose a t-order input-memory nonlinear finite automaton $M'_1 = \langle Y, Y, S'_1, \delta'_1, \lambda'_1\rangle$ defined by (1), a finite automaton $M_1 = \langle Y, Y, S_1, \delta_1, \lambda_1\rangle$ and a finite automaton $M''_1 = \langle Y, Y, S''_1, \delta''_1, \lambda''_1\rangle$ such that M'_1 is a weak inverse with delay τ of M_1 and M''_1 is a weak inverse with delay τ' of M'_1. (Such M_1, M'_1 and M''_1 are existent, for example, see [16] pp. 182—183.) 4) From M'_0 and M'_1, make the finite automaton $C'(M'_1, M'_0)$ expressed in the form of (12). 5) Choose a state s_1 of M_1, a state $s'_1 = \langle y(-t), \cdots, y(-1)\rangle$ of M'_1 and a state s''_1 of M''_1 such that s'_1 and s_1 is a match pair with delay τ and s''_1 and s'_1 is a match pair with delay τ'. Choose $y(-r-t), \cdots, y(-t-1)$ in Y, and let $s'_0 = \langle y'(-r), \cdots, y'(-1)\rangle$, where $y'(i) = f(y(i-t), \cdots, y(i))$, $i = -1, \cdots, -r$. Then $(C'(M'_1, M'_0), \langle y(-r-t), \cdots, y(-1)\rangle, r+\tau, r+\tau')$ and $(M_0, M'_0, M_1, M''_1, s'_0, s_1, s''_1)$ are the public key and the secret key of user A, respectively.

When another user B wishes to send a message $y(0)\cdots y(n)$ to user A in secrecy, B first extends arbitrary $r+\tau'$ digits $y(n+1), \cdots, y(n+r+\tau')$ in Y, then calculates the ciphertext $x'(0)\cdots x'(n+r+\tau')$ according to (12) using A's public key, which is sent to user A. On receipt, user A first calculates $\lambda_0(0, x'(0)\cdots x'(n+r+\tau') - \lambda'_0(s'_0, 0^{n+r+\tau'+1}))$ denoted by $y''(0)\cdots y''(n+r+\tau')$ using M_0, M'_0 and s'_0 in A's secret key, then calculates $\lambda''_1(s''_1, y''(r)\cdots y''(n+r+\tau'))$ using M''_1 and s''_1 in A's secret key which is equal to $\bar{y}(0)\cdots\bar{y}(\tau'-1)y(0)\cdots y(n)$ for some $\bar{y}(0), \cdots, \bar{y}(\tau'-1)$ in Y.

To prove $\lambda''_1(s''_1, y''(r)\cdots y''(n+r+\tau')) = \bar{y}(0)\cdots\bar{y}(\tau'-1)y(0)\cdots y(n)$ for some $\bar{y}(0), \cdots, \bar{y}(\tau'-1)$ in Y, we denote $\lambda'_1(s'_1, y(0)\cdots y(n+r+\tau')) = y'(0)\cdots y'(n+r+\tau')$. From Proposition 2, we have $\lambda'_0(s'_0, y'(0)\cdots y'(n+r+\tau') = x'(0)\cdots x'(n+r+\tau')$. Since M'_0 is linear, we have $x'(0)\cdots x'(n+r+\tau') - \lambda'_0(s'_0, 0^{n+r+\tau'+1}) = \lambda'_0(0, y'(0)\cdots y'(n+r+\tau'))$. Since M'_0 is an inverse with delay r of M_0 and they are linear, we have $y''(0)\cdots y''(n+r+\tau') = \lambda_0(0, \lambda'_0(0, y'(0)\cdots y'(n+r+\tau')))$ $= 0^{r} y'(0)\cdots y'(n+\tau')$. It follows immediately that $y''(r)\cdots y''(n+r+\tau') = y'(0)\cdots y'(n+\tau')$. Since s''_1 and s'_1 is a match pair with delay τ', we have $\lambda''_1(s''_1, y''(r)\cdots y''(n+r+\tau')) = \lambda''_1(s''_1,\ \lambda'_1(s'_1,\ y(0)\cdots y(n+\tau'))) = \bar{y}(0)\cdots\bar{y}(\tau'-1)y(0)\cdots y(n)$ for some $\bar{y}(0), \cdots, \bar{y}(\tau'-1)$ in Y.

This public key cryptosystem can be used to implement digital signatures. User A signs a message $x(0)\cdots x(n)$ as follows. 1) User A first extends arbitrary $r+\tau$ digits $x(n+1),\cdots,x(n+r+\tau)$ in X. 2) Then A chooses any state s_0 of M_0 and calculates $y(0)\cdots y(n+r+\tau)=\lambda_1(s_1,\lambda_0(s_0,x(0),\cdots,x(n+r+\tau)))$ using M_0, M_1 and s_1 in A's secret key. Every one, say B, validates A's signature $y(0)\cdots y(n+r+\tau)$ on $x(0)\cdots x(n)$ as follows. 1) User B first chooses arbitrary $\bar{y}(-r-t),\cdots,\bar{y}(-t-1)$ in Y. 2) Then B calculates $x'(0)\cdots x'(n+r+\tau)=\lambda'(\langle\bar{y}(-r-t),\cdots,\bar{y}(-t-1),y(-t),\cdots,y(-1)\rangle, y(0)\cdots y(n+r+\tau))$ using A's public key, where λ' is the output function of $C'(M_1', M_0')$. From Theorem 2 and its proof, it is easy to see that $x'(r+\tau)\cdots x'(r+\tau+n)=x(0)\cdots x(n)$.

4. Security

For the varieties of finite automaton public key cryptosystem stated in previous section, their security may be analogously discuss as in [1]. As pointed out there, the security of these cryptosystems is determined by the complexity of finding weak inverse finite automaton of $C'(M_0, M_1)$ or of $C'(M_1', M_0')$ (of finding weakly invertible finite automaton with weak inverse $C'(M_0, M_1)$ or $C'(M_1', M_0')$ for digital signatures). But the mathematics does not yet provide a systematic method to estimate the precise lower bounds of computing time and storage amount for finding weak inverses of nonlinear finite automata (for finding weakly invertible finite automata of which a given nonlinear finite automaton is a weak inverse). So the only way available is to design a good algorithm and to estimate the precise upper bound of computing time and storage amount which is regarded as a loose approximation of the lower bound.

For the sake of avoiding repetition, we only discuss the problem for the case in §3.2, the case in §3.3 can be analogously discussed. For finding a weak inverse with delay τ of $C'(M_1', M_0')$, the first method is a general one which is fit to any weakly invertible finite automata with delay τ. Denote the output function of $C'(M_1', M_0')$ by λ'. Suppose that $C'(M_1', M_0')$ is weakly invertible with delay τ' and $\tau'\leqslant\tau$. Since $C'(M_1', M_0')$ is an input-memory finite automaton, it is strongly connected. For finding a weak inverse with delay τ' of $C'(M_1', M_0')$, according to this method, we need for each state $s'=\langle y(-r-t),\cdots,y(-1)\rangle$ of $C'(M_1', M_0')$ to calculate all $x(0)\cdots x(\tau')=\lambda'(s', y(0)\cdots y(\tau'))$ for $y(0),\cdots,y(\tau')$ in Y. from which a function f can be deduced with $f(s', x(0)\cdots x(\tau'))=y(0)$. Since the state number of $C'(M_1', M_0')$ is $q^{l(r+t)}$ and there are $q^{l(\tau'+1)}$ sequences over Y with length $\tau'+1$, we need to calculate $q^{l(r+t+\tau'+1)}$ values of function λ'. In case of $q=2$, $l=8$, $r=t=10$ and $\tau'=0$, we have $q^{l(r+t+\tau'+1)}=2^{168}>10^{50}$! Hence, this method is impractical for moderate $r+t+\tau'$.

For finding an automaton of which $C'(M_1', M_0')$ is a weak inverse, a general method is given in [22]. This method spends more computing time and storage amount than the general method above.

The second method is a special one for $C'(M_1', M_0')$. The centre of this method is to decompose $C'(M_1', M_0')$. M_0 and M_1 can be easily found out as soon as M_0' and M_1' are obtained from decomposing $C'(M_1', M_0')$. Since $C'(M_1', M_0')$ is given by (12) and the coefficients in (12) satisfy (13), decomposition of $C'(M_1', M_0')$ is equivalent to factorization of matrix polynomials over $GF(q)$. Since the matrix ring over $GF(q)$ is noncommutative and contains divisors of zero, establishing divisibility theory for matrix polynomials over $GF(q)$ with singular leading coefficient seems rather difficult [23]. Although polynomial time algorithms for factorization of polynomials over $GF(q)$ are existent, for example, see [24, 25], yet no feasible algorithm exists for factoring matrix polynomials over $GF(q)$. A possible straight way is to reduce (13) to a simultaneous quadratic equation over $GF(q)$ and solve it. But it is well known that solving nonlinear equations over $GF(q)$ is very difficult if its argument number is great.

From above discussion, security is relative to the size of parameters q, i, r, t and τ'. Usually, $q = 2$. We recommend $i \geqslant 8$, $r \geqslant 10$ and $t \geqslant 10$. And nonlinear function f may be chosen such that its polynomial expression (10) contains a few monomials with span $\geqslant 2$. For example, take f as the following form:

$$f(y_{-t}, \cdots, y_0) = F_0 + \sum_{j=0}^{t} F_{jj} y_{-j} + \sum_{j=0}^{t-1} F'_{j,j+1} y_{-j} y_{-j-1}, \tag{16}$$

where F_0, F_{jj} and $F'_{j,j+1}$ are $l \times 1$, $l \times l$ and $l \times l$ matrices over $GF(q)$, and $y_{-j} y_{-j-1} = [a_1 b_1, \cdots, a_l b_l]^T$ for any $y_{-j} = [a_1, \cdots, a_l]^T$ and $y_{-j-1} = [b_1, \cdots, b_l]^T$. In this instance, (12) is simplified as the following:

$$x'(i) = C_0 + \sum_{j=0}^{r+t} C_{jj} y(i-j) + \sum_{j=0}^{r+t-1} C'_{j,j+1} y(i-j) y(i-j-1),$$

$$i = 0, 1, \cdots, \tag{17}$$

where

$$\begin{aligned} C_0 &= \sum_{j=0}^{r} B_j F_0, \\ C(z) &= B(z) F(z), \\ C'(z) &= B(z) F'(z), \\ B(z) &= \sum_{j=0}^{r} B_j z^j, \\ F(z) &= \sum_{j=0}^{t} F_{jj} z^j, \\ F'(z) &= \sum_{j=0}^{t-1} F'_{j,j+1} z^j, \\ C(z) &= \sum_{j=0}^{r+t} C_{jj} z^j, \\ C'(z) &= \sum_{j=0}^{r+t-1} C'_{j,j+1} z^j. \end{aligned} \tag{18}$$

Since $C'(M'_1, M'_0)$ in the public key is given by coefficients C_0, C_{ii}, $i = 0, \cdots, r + t$, and $C'_{i,i+1}$, $i = 0, \cdots, r + t - 1$, both the lengths of the public keys in two varieties are about $[l(1 + l(2r + 2t + 1)) + l(r + t)] \log_2 q$ bits. Letting $q = 2$ and $l = 8$, the length of public key is about $349 \times 8 = 2792$ bits in case of $r = t = 10$, $519 \times 8 = 4152$ bits in case of $r = t = 15$, and $689 \times 8 = 5512$ bits in case of $r = t = 20$.

References

[1] Tao Renji and Chen Shihua, A finite automaton public key cryptosystem and digital signatures, *Chinese J. ofComputer*, 8(1985), 401—409.

[2] W.Diffie and M.Hellman, New directions in cryptography, *IEEE Trans. Inform. Theory*, **22**(1976), 644—654.

[3] R.C.Merkle and M.E.Hellman, Hiding information and signatures in trapdoor knapsacks, *IEEE Trans. Inform.Theory*, **24**(1978), 525—530.

[4] M.Willett, Trapdoor knapsacks without superincreasing structure, *Inform. Proc. Letters*, **17**(1983), 7—11.

[5] A.Shamir, Embedding cryptographic trapdoors in arbitrary knapsack systems, *Inform. Proc. Letters*, **17**(1983), 77—79.

[6] R.Cooper and W.Patterson, A generalization of the knapsack algorithm using Galois fields, *Cryptologia*, **8**(1984),343—347.

[7] A.Shamir, A polynomial time algorithm for breaking Merkle-Hellman crytosystems, Proc. of the 23rd Annual Symp.Zon the Foundations of Computer Science, 1982, 145—152.

[8] L.M.Adleman, On breaking generalized knapsack public key cryptosystem, Proc. of the 15th Annual ACM Symp. onTheory of Computing, 1983, 402—412.

[9] R.L.Rivest, A.Shamir and L.Adleman, A method for obtaining digital signatures and public key cryptosystems,*Comm. ACM*, **21**(1978), 120—126.

[10] W.B.Müller and W.Nöbauer, Some remarks on public-key cryptosystems, *Studia Sci. Math. Hung.*, **16**(1981), 71—76.

[11] H.Brändström, A public-key cryptosystem based upon equations over a finite field, *Cryptologia*, **7**(1983), 347—358.

[12] T.ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, Crypto '84.

[13] M.R.Magyarik and N.R.Wagner, A public-key cryptosystem based on the word problem, Crypto '84.

[14] R.J.McEliece, A public-key cryptosystem based on algebraic coding theory, DSN Progress Report, 42—44, 1978.

[15] Zhou Tongheng, Boolean public key cryptosystem of the second order, J. of China Inst. of Communications, **5**(1984),30—37.

[16] Tao Renji, Invertibility of finite automata, Science Press, Beijing, 1979(in Chinese).

[17] Tao Renji and Chen Shihua, Some properties on the structure of invertible and inverse finite automata with delay τ,*Chinese J. of Computer*, **3**(1980), 289—297.

[18] Chen Shihua, On the structure of weak inverses of a weakly invertible linear finite automaton, *Chinese J.of Computer*, **4**(1981), 409—419.

[19] Tao Renji, Relationship between bounded error propagation and feedforward invertibility, KEXUE TONGBAO,27(1982), 680—682.

[20] Tao Renji, Some results on the structure of feedforward inverses, *Scientia Sinica*, ser.A, **27**(1984), 157—162.

[21] Chen Shihua, On the structure of (weak) inverses of an (weakly) invertible finite automaton, *J.of Computer Science and Technology*, 3(1986) (to appear).

[22] Chen Shihua, On the structure of finite automata of which M' is an (weak) inverse with delay τ, *J. of Computer Science and Technology*, 2(1986) (to appear).

[23] I.Gohberg, P.Lancaster and L.Rodman, Matrix Polynomials, Academic Press, New York, 1982.

[24] E.Berlekamp, Algebraic coding Theory, McGraw-Hill Book Co., New York, 1968.

[25] E.Berlekamp, Factoring polynomial over large finite fields, *Math. Comp.*, **24**(1970), 713—735.