Digital Signature Schemes Based on LFSR Sequences

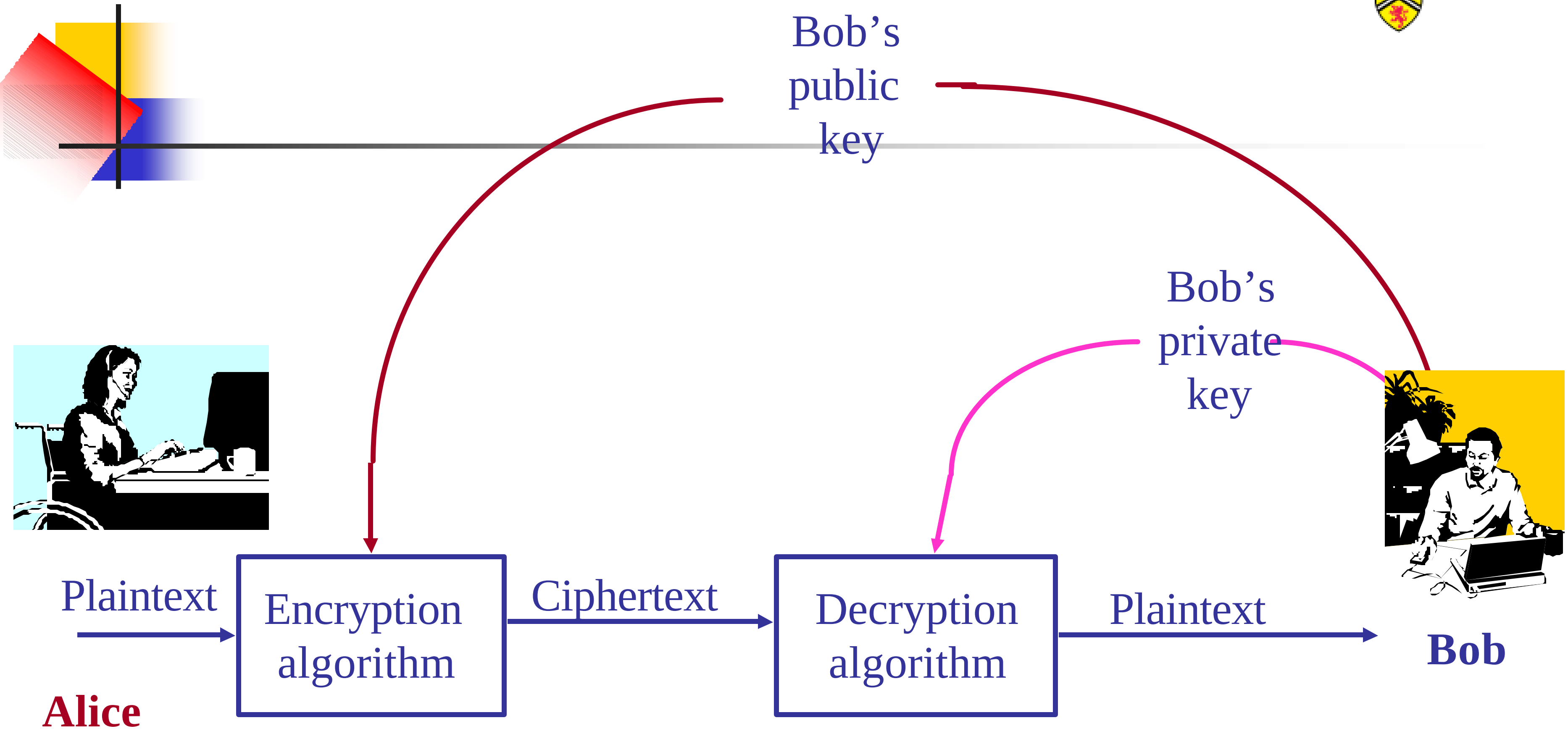
Guang Gong
Department of Electrical & Computer Engineering
University of Waterloo, CANADA

Presentation Outline

- Overview of Digital Signature Schemes
- Characteristic Sequences over GF(q) of Degree n and Commutative Law
- Digital Signature Schemes Based on Characteristic Sequences and the Trace-Discrete-Logarithm
- Efficient Digital Signature Schemes Based on the Sequences for $n = 3$ and $n = 5$
- Related work: LUC, XTR and Toris Based Cryptography

Overview of Digital Signature Schemes

- Basics of Public-key Cryptography
- RSA Encryption and Digital Signature
- ElGamal Digital Signature and DSS (Digital Signature Standard)
- ECDSA (Elliptic Curve Digital Signature Algorithm)



Simplified Model of Public-Key Encryption

Requirements of Public-key Cryptography

One-way function:

$$x \xrightarrow{\text{easy}} f(x)$$

$$x \xleftarrow{\text{infeasible}} f(x)$$

Trapdoor one-way function:

$$x \xrightarrow{\text{easy}} f_k(x)$$

$$x \xleftarrow[\text{easy if } k \text{ is known}]{\text{infeasible if } k \text{ is not known}} f_k(x)$$

Therefore, security of public-key cryptosystems are based on the difficulty of different computational problems.

Most important ones are

- Factoring large integers
- Finite field discrete logarithms
- Elliptic curve discrete logarithms

Key pairs of the public-key system

In a secure network system, each user x has a pair of keys (E_x, D_x):

- E_x is an encryption key which is put into a public key directory or a file (after certified), called a public-key of the user.
- D_x is a decrypted key kept private, called a private key of the user.
- $D_x(E_x) = E_x(D_x) =$ identity map
- From known E_x, it is computational infeasible to obtain D_x

$$\text{Alice} \xrightarrow{\quad C = E_b(m) \quad} \text{Bob: } D_b(C) = D_b E_b(m) = m$$

Requirements of Digital Signatures

- Everyone can verify digital signatures.
- Only the signer can sign; no one can forge the signer's signature (this prevents forgery and denial attacks.)
- Once a dispute occurs, a third party can solve it.

RSA Digital Signature Algorithm (RSA-DSA)

Signer: - Select p and q both prime; $n = pq$; e: $\gcd(e, f(n)) = 1$, $1<e< f(n)$.
Compute: $d = e^{-1} \bmod f(n)$.
Public key: $\{e, n\}$. Private key: $\{d, p, q\}$
- $h(.)$: a hash function (e.g. SHA-1)

Signer

- Computes $h(m)$ and

$$r = h(m)^d \bmod n$$

r is a digital signature of the message m

Verifier

- computes $r^e \bmod n$
- checks whether

$$r^e = h(m) \qquad (1)$$

If (1) is true, accepts as a valid signature. Otherwise, rejects it.

Remark: Most frequently used in wireless communications since e can be chosen as 3 which extremely saves the cost of the verification process.

ElGamal Digital Signature Algorithm (1985) and Digital Signature Standard (DSS) (NIST, 1994)

- System public keys: p, a prime, Q, a factor of $p-1$, g an element in GF(p) with order Q
- $h(.)$: a hash function
- Signer, private key: $0 < x < Q$ with $(x, Q) = 1$, public key: $y = g^x$.

Signing

- randomly picks k: $0 < k < Q$ coprime with Q (per message)
- computes $r = g^k$
- solves for t in the equation:

  $h(m) \equiv xr + kt \pmod{Q}$

(r, t) is a digital signature of the message m

Verifying

- setting $u = h(m)t^{-1} \bmod Q$

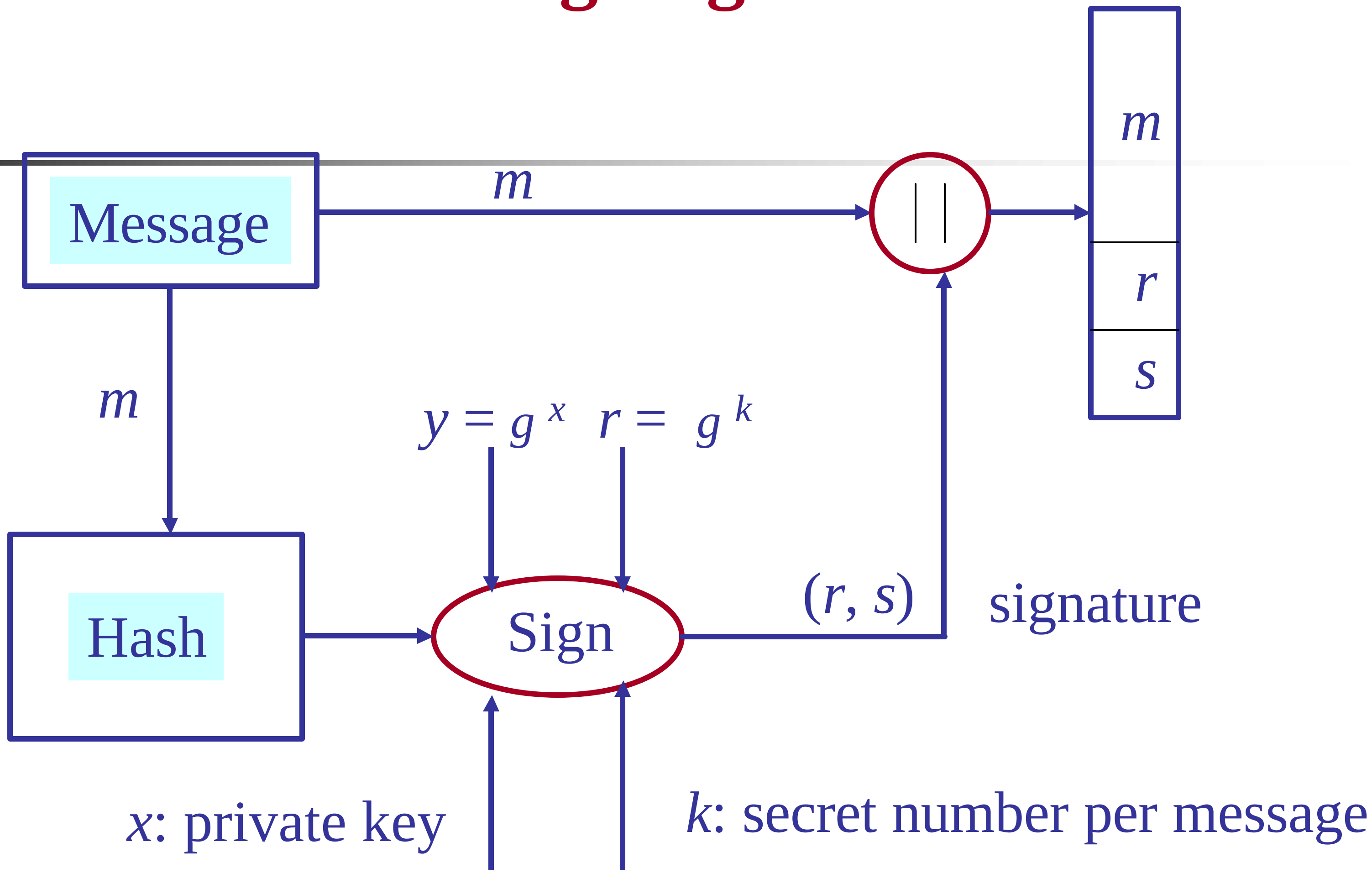
  $v = -r\,t^{-1} \bmod Q$
- computes $w = g^u y^v$
- checks whether

  $w = r \qquad (1)$

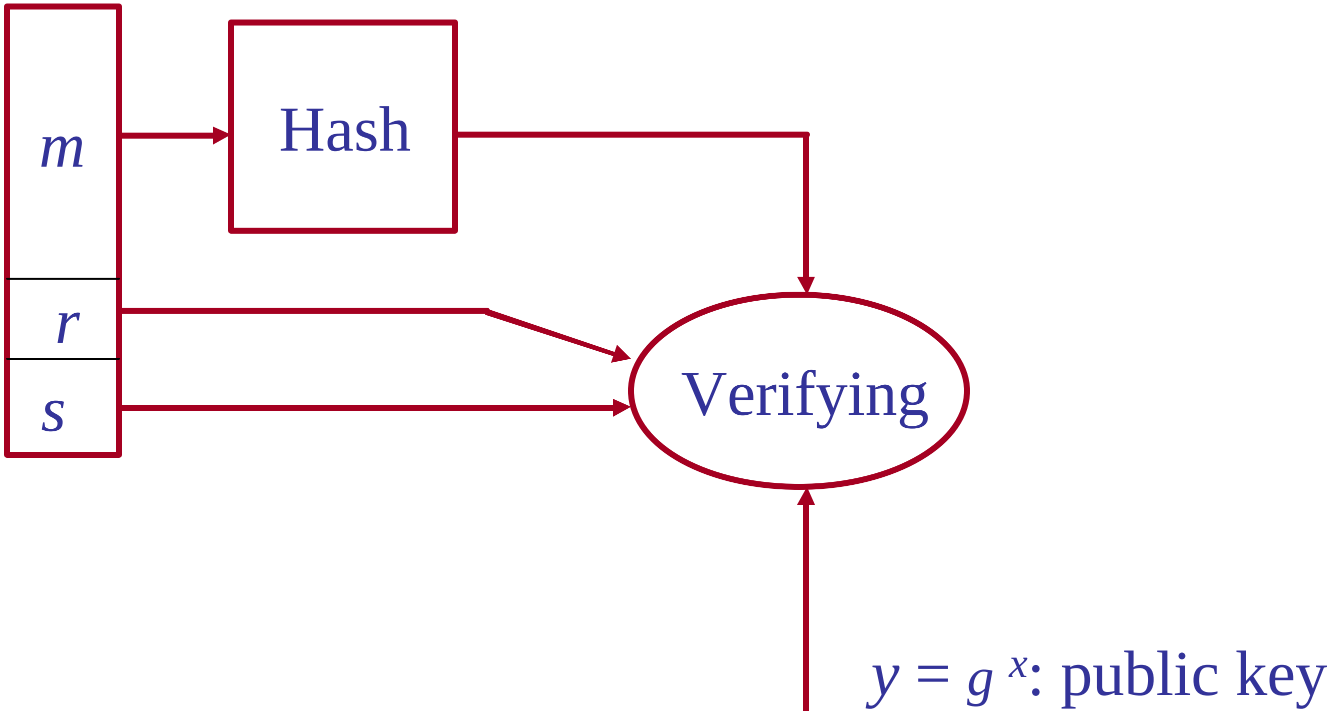
If (1) is true, accept as a valid signature. Otherwise, reject it.

In ElGmal, $Q = p - 1$, and in DSS, Q is a 160 bit number. In elliptic curve digital signature algorithm (EC-DSA), g is replaced by a point on an elliptic curve, and the multiplicative group of GF(p) is replaced by an additive group of points on the curve. But the signing equation and all the procedures are preserved.

ElGamal and DSS Signing Process



ElGamal and DSS Verifying Process



Security of the ElGamal-like Signature Scheme

Consider

$$m = xr + ks \bmod p-1 \qquad (1)$$

If the attacker can compute $y = g^x$ to obtain x, then he can forge any signature since in (1) he can pick k to compute r, and therefore, obtain s.

Thus the security of the ElGamal digital signature algorithm is based on the difficulty of solving discrete log problem in F_p .

Remark: The signing equation (1) can be changed to other forms. We will refer to all signature schemes using the ElGamal procedure with a different signing equation, or different group, or different order of g, as ElGamal-like signature schemes.

Characteristic Sequences over GF(q) of Degree n and Commutative Law

- Let q be a prime or a power of a prime,

$$f(x) = x^n - a_{n-1}x^{n-1} + a_{n-2}x^{n-2} - \cdots + (-1)^{n-1}a_1 x + (-1)^n, \quad a_i \in GF(q)$$

irreducible over GF(q) with order Q,, and let a be a root of $f(x)$ in the extension GF(q^n).

- A sequence $\mathbf{s} = \{s_k\}$ is said to be an LFSR sequence generated by $f(x)$ if

$$s_{k+n} = a_{n-1}s_{k+n-1} + a_{n-2}s_{k+n-2} - \cdots + (-1)^{n-1}a_1 s_{k+1} + (-1)^n s_k, \; k = 0,1,\cdots$$

- If an initial state of $\{s_k\}$ is given by

$$s_k = Tr(a^k), \; k = 0,1,\cdots,n-1$$

then $\{s_k\}$ is called a (nth-order) *characteristic sequence*.

We denote $s_k = s_k(f)$, $k = 0, 1, \ldots$.

Characteristic Sequences of Degree 3

- Let q be a prime or a power of a prime and

$$f(x) = x^3 - a x^2 + bx - 1, a, b \in GF(q),$$

be irreducible over $GF(q)$.

- A sequence $\{s_k\}$ is said to be an LFSR sequence generated by $f(x)$ if

$$s_{3+k} = as_{2+k} + bs_{1+k} + s_k, k = 0, 1, \ldots$$

- If an initial state of $\{s_k\}$ is given by

$$s_0 = 3, s_1 = a, \text{ and } s_2 = a^2 - 2b,$$

then $\{s_k\}$ is called a (3rd-order) *characteristic sequence*.

Example 1. Let K = GF(5), r = 3 and

$f(x) = x^3 + x - 1$ which is irreducible over K.

The characteristic sequence generated by $f(x)$:

3	0	3	3	2	0	1	2	4	4
3	0	1	3	4	3	4	1	4	3
2	1	1	1	0	0	1	0	4	1
1	...								

which has period $31 = 5^2 + 5 + 1$.

The reciprocal polynomial of $f(x)$ is

$$f^{-1}(x) = x^3 - x^2 - 1$$

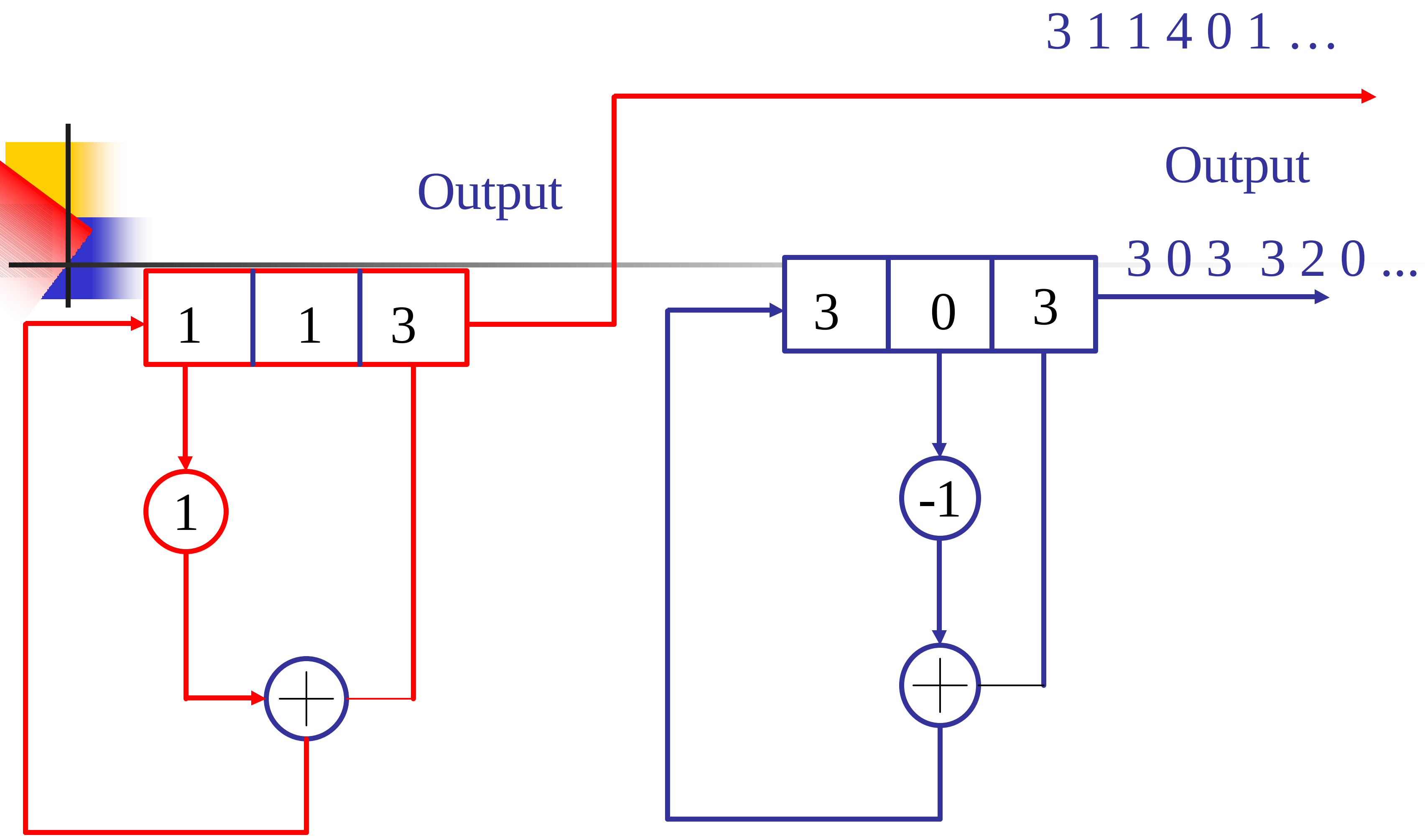
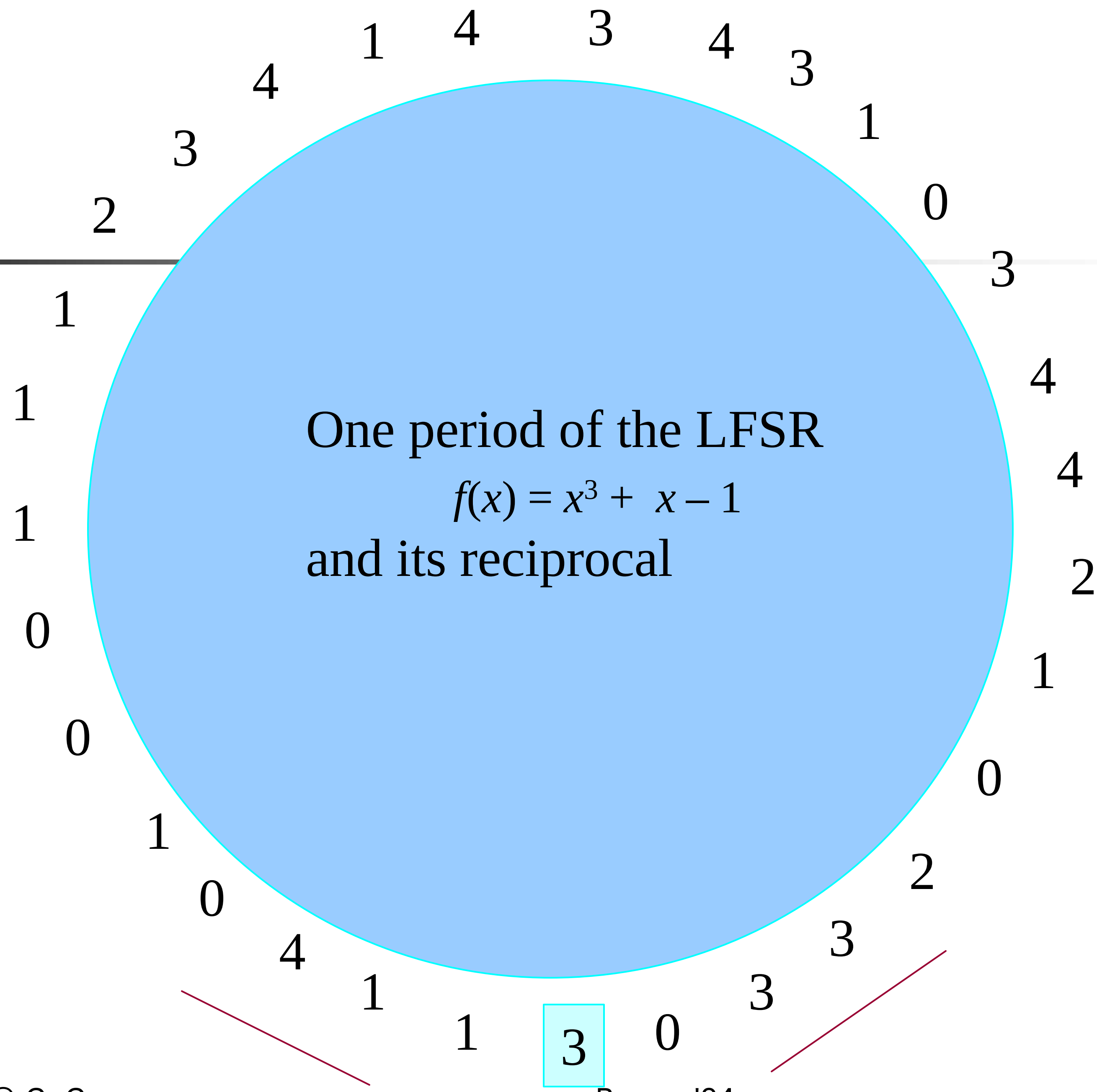


Figure 2. A Pair of the Reciprocal LFSRs in Example 1


One period of the LFSR
$f(x) = x^3 + x - 1$
and its reciprocal
3 4 3 1 0 3 4 4 2 1 0 2 3 3 0 3 1 1 4 0 1 0 0 1 1 1 2 3 4 1 4

Profiles of *n*th-order Characteristic Sequences

- Period : a factor of $q^{n-1} + \ldots + q + 1$
- Trace representation:

$$s_k = Tr(\alpha^k) = \alpha^k + \alpha^{kq} + \cdots + \alpha^{kq^{n-1}}, \quad k = 0,1,\ldots$$

- For any two positive integers k and e, let $f_k(x)$ be the minimal polynomial of α^k over GF(q). Then

$$s_e(f_k) = s_{ek}(f) = s_k(f_e)$$

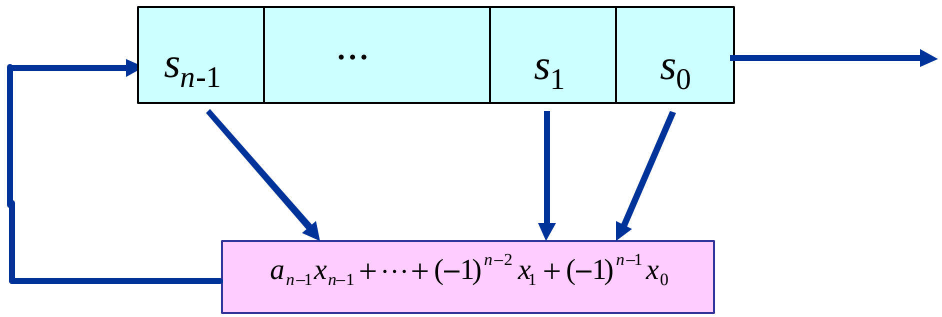
which is called the commutative law of the char. sequences.

- Let

$$f_k(x) = x^n - a_{n-1,k} x^{n-1} + \cdots + (-1)^{n-1} a_{1,k} + (-1)^n$$

Then $s_k = a_{n-1,k}$ and $s_{-k} = a_{1,k}$

State Transition of LFSR Sequences



- Let $\{s_k\}$ be generated by $f(x)$,
- State vector:

$$\mathbf{s}_j = (s_j, s_{j+1}, \cdots, s_{j+n-1})$$

- State transition matrix:

$$A = \begin{bmatrix} 0 & 0 & \cdots & 0 & (-1)^{n-1} \\ 1 & 0 & & 0 & (-1)^{n-2}a_1 \\ 0 & 1 & & 0 & (-1)^{n-3}a_2 \\ \cdots & & & & \\ 0 & 0 & & 1 & a_{n-1} \end{bmatrix}$$

State transition formulas:

$$\begin{aligned} \mathbf{s}_j &= (s_{j-1}, s_j, \cdots, s_{j+n-2})A \\ &= \cdots \\ &= (s_0, s_1, \cdots, s_{n-1})A^j \end{aligned}$$

Let

$$M(j) = \begin{bmatrix} \mathbf{s}_j \\ \mathbf{s}_{j+1} \\ \vdots \\ \mathbf{s}_{j+n-1} \end{bmatrix}$$

Property 1. $\mathbf{s}_{v+j} = \mathbf{s}_v(M(0)^{-1}M(j))$

Therefore, the $(v+j)$th term, s_{v+j}, is the inner product of $\mathbf{s}_v$ and the first column of $M(0)^{-1}M(j)$.

Motivation of the LFSR based public-key cryptography

- Develop a PKC whose security is based on the difficulty of solving the discrete logarithm (DL) in GF(q^n), but all computation are performed in GF(q).

One important issue needs to be solved:

Fast computation algorithm for evaluating s_k, the k^{th} term of the sequence.

If we can find an algorithm which computes the k^{th} term of s is faster than to compute a^k in GF(q^n) for some n, then we can have an efficient digital signature scheme.

Algorithms for kth Term Computation of Char. Sequences

Assumption A (kth Term Computation). For a given $f(x)$ and k, there is an efficient algorithm to calculate the kth term s_k (compare it with calculating a^k. This will become Algorithms for $n = 3$ and $n = 5$.)

Algorithm 1. (Mixed Term Computation) For given v and $\mathbf{s}_j = (s_j, s_{j+1,} \cdots, s_{j+n-1})$ (j is unknown), the (v+j)th term, s_{v+j}, can be computed by the following procedure:

Step 1. Compute the vth state, $\mathbf{s}_{v,}$ using Assumption A.

Step 2. Compute s_{v+j} by Property 1, which only involves matrix computation.

ElGamal-like Digital Signature Algorithm Based on LFSR Sequences

-System public keys: $f(X)$ with the constant term $(-1)^n$, an irreducible polynomial over GF(q) of degree n with period Q; a a root of $f(x)$ in GF(q^n).
- $h(.)$: a hash function.
- Signer, private key: $0< x < Q$ with $(x, Q) = 1$, public key: $\mathbf{y} = \mathbf{s}_x$, the xth state of the char. sequence of $f(x)$.

Signing

- randomly picks k: $0 < k < Q$ coprime with Q (per message)
- computes f_k, as a vector of $n - 1$ dimensional space, the minimal polynomial of a^k over GF(q), set r is an integer converted from s_k.
- solves for t in the equation:
  $h(m) \equiv xr + kt \pmod Q$

(f_k, t) is the digital signature of the message m.

Verifying

- setting $v = - h(m)r^{-1} \bmod Q$
  $u = - t\, r^{-1} \bmod Q$
- Using Algorithm 1 to compute $A = s_{v+x}$ from v and $\mathbf{s}_x$.
- Using Assumption A to compute $B = s_u(f_k)$, the uth term of the char. sequence of f_k.
- checks whether
  $A = B \qquad (1)$

If (1) is true, accept as a valid signature. Otherwise, reject it.

Signer: x

$\mathbf{s}_x = (s_x, s_{x+1}, \cdots, s_{x+n-1})$

Signing for the message m:

k

$f_k = (s_k, a_{k,n-2}, \cdots, a_{k,2}, s_{-k})$

$r = \text{convert}(s_k)$

x, k, m → $h(m) \equiv xr + kt \pmod{Q}$

t

Signature: (r, t). (f_k transmitted).

Verifier:

m, r → $v = -h(m)r^{-1}$

t, r → $u = -tr^{-1}$

$\mathbf{s}_x$ → $A = s_{v+x}$ (Al 1)

f_k → $B = s_u(f_k)$(Assu)

$A = ?\, B$

Yes, accept it.

No, reject it.

Sign and Verifi. of ElGamal-like LFSR-DSA

Security of ElGamal-like LFSR-DSA and the Trace Discrete Logarithm Problem

How to forge a signature of the LFSR-DSA?

- There is a one-to-one correspondence between the set consisting of all states of the LFSR $f(x)$ and the set consisting of all powers of α (a root of $f(x)$), which is the subgroup of the multiplication group of GF(q^n).

So, if one solves the discrete logarithm problem (DLP) in a polynomial time (i.e., given α and β, solving for d such that $b = a^d$), then from the public-key $\mathbf{s}_x$, through the above one-to-one correspondence, the attacker can obtain the private key x. From this, he (she) can forge any signature as wish.

Security of ElGamal-like LFSR-DSA and the Trace Discrete Logarithm Problem (Cont.)

Definition. Given $b \in GF(q)$, the trace discrete logarithm problem is of finding an index *d* such that $Tr(a^d) = b$, or determining there is no such index .

If one cannot solves the DLP in a polynomial time, but can solve the trace-DLP in a polynomial time, then there is a forgery as follows. Let *m* be the message that the attacker wishes to forge a signature.

The attacker may:

(1) randomly choose *k*, and compute f_k by Assumption 1, so *r* is obtained.

(2) compute $A = s_{v+x}$ where $v = -h(m)r^{-1}$ by Al 1.

(3) find an index *d* by solve the trace-DLP for $A = Tr(a^d)$.

(4) set $t = -rdk^{-1} \pmod{Q}$

Then (f_k, t) is a forged signature of *m*.

Result. The security of the LFSR-DSA is based on the difficulty to solve the trace-DLP.

Question. What is the relationship between the complexity of the DLP and the complexity of the trace-DLP? The answer is that they are equivalent under some assumption (omitted here).

Efficient Digital Signature Algorithm Based on LFSR Sequences of Degrees 3 and 5

- Fast Algorithm to Evaluate the *k*th term of Char. Sequences of Degree 3
- Cubic DSA and Applications in the Constrained Devices
- Fast Algorithm to Evaluate the *k*th Term of Char. Sequences of Degree 5
- Quintic DSA

Fast algotihm to evaluate the *k*th term of char. Sequences with degree 3

Let $\{s_k\}$ be the characteristic sequence over GF(q) generated by

$$f(x) = x^3 - ax^2 + bx - 1$$

and $\{s_{-k}\}$, generated by the reciprocal of $f(x)$, which is given by

$$f^{-1}(x) = x^3 - bx^2 + ax - 1$$

Lemma 1. For any two integers n and m, we have

(1) $s_{2n} = s_n - 2s_{-n}$

(2) $s_n s_m - s_{n-m} s_{-m} = s_{n+m} - s_{n-2m}, \quad n \neq m$

From this lemma, we can obtain an algorithm to compute the kth state and its reciprocal state, therefore, the kth term of the sequence.

Algorithm 2 (Reciprocal States Fast Evaluation Algorithm (RSEA), Gong-Harn, 1999)

Let $k = \sum_{i=0}^{r} k_i 2^{r-i}$ be the binary representation of *k*. Let $T_0 = k_0 \neq 0$ and $T_j = k_j + 2T_{j-1}$, $1 \leq j \leq r$. So, $T_r = k$. Then the *k*th terms of a pair of the reciprocal char. sequences can be computed iteratively as follows:

For $k_j = 0$,

$$s_{T_j-1} = s_{T_{j-1}} s_{T_{j-1}-1} - b s_{-T_{j-1}} + s_{-(T_{j-1}+1)},$$

$$s_{T_j} = s_{T_{j-1}}^2 - 2 s_{-T_{j-1}}, \text{ and}$$

$$s_{T_j+1} = s_{T_{j-1}} s_{T_{j-1}+1} - a s_{-T_{j-1}} + s_{-(T_{j-1}-1)}.$$

For $k_j = 1$,

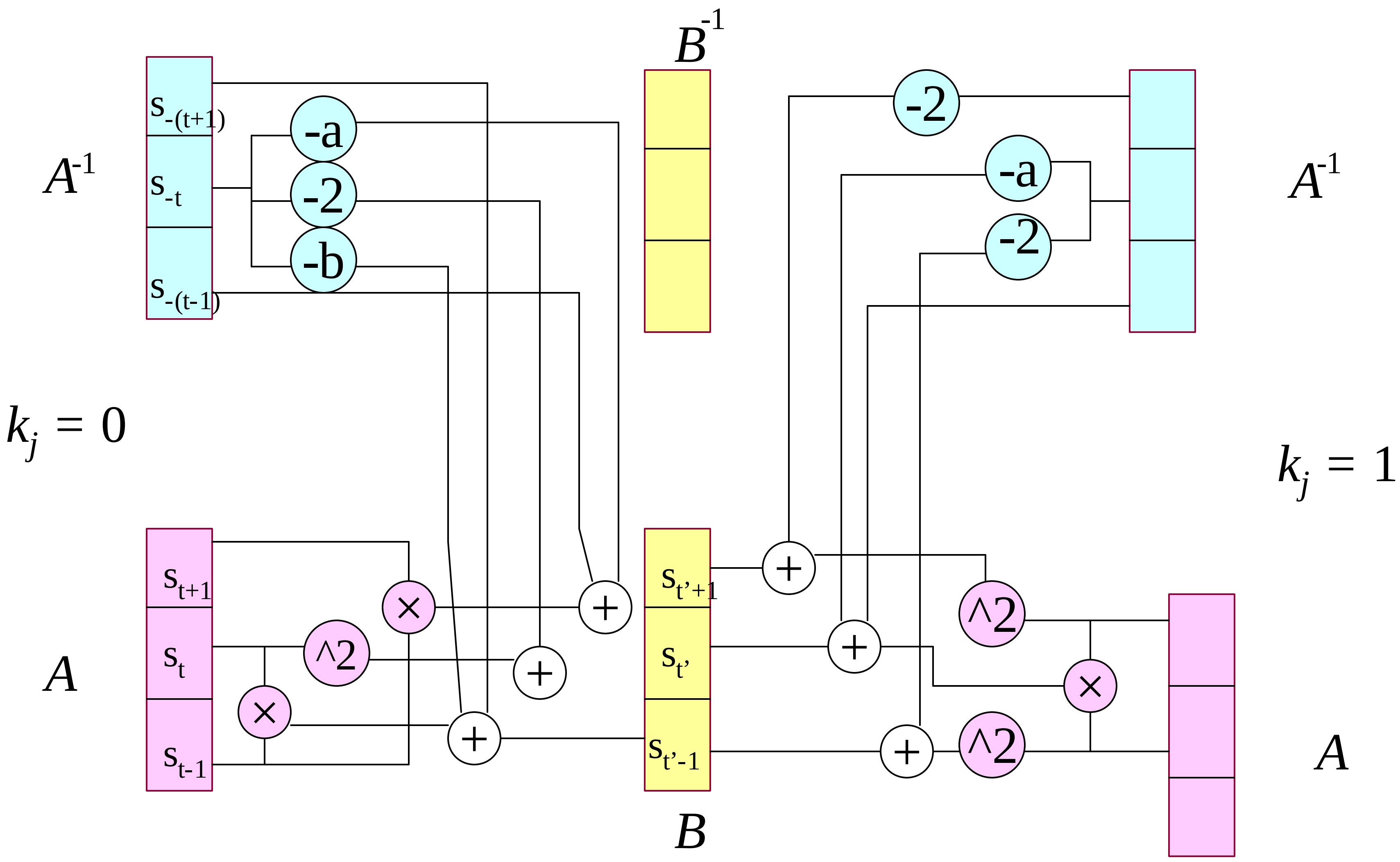
$$s_{T_j-1} = s_{T_{j-1}}^2 - 2 s_{-T_{j-1}},$$

$$s_{T_j} = s_{T_{j-1}} s_{T_{j-1}+1} - a s_{-T_{j-1}} + s_{-(T_{j-1}-1)}, \text{ and}$$

$$s_{T_j+1} = s_{T_{j-1}+1}^2 - 2 s_{-(T_{j-1}+1)}.$$

Thus evaluation of a pair of the *k*th terms s_k and s_{-k} needs $9\log k$ multiplications in GF(q) in average.

RSEA outputs dual states of the LFSR $f(x)$



$t = T_{j-1}$ and $t' = T_j = k_j + 2T_{j-1}$

Redundancy identities of States of the 3rd-Order Characteristic Sequences

- Reciprocal operator: $D(s_k) = s_{-k}$
- For given reciprocal terms (s_k, s_{k+1}) and $(s_{-k}, s_{-(k+1)})$,

  if $D = s_{k+1} s_{-(k+1)} - ab \neq 0$, then

$$s_{k-1} = (e\, s_{-(k+1)} - b\, D(e))/D \quad \text{and} \quad s_{-(k-1)} = D(s_{k-1})$$

$$\text{where } e = s_k^2 + (ab - 3)\, s_{-k} - a\, s_{-(k+1)}$$

- This shows that three elements in any state of the 3rd-order characteristic sequences are not independent.

Algorithm 3. An Algorithm for Computing a Mixed Term s_{v+j}, j Unknown (Algorithm 1 in Degree 3 version)

Input: v and $\mathbf{s}_j = (s_j, s_{j+1}, s_{j+2})$.

Output: s_{v+j}, the $(v+j)$th term of the Char. Sequence.

Procedure:

Step 1: Applying Algorithm 2 to compute $\mathbf{s}_v = (s_v, s_{v+1}, s_{v+2})$, the vth state of the LFSR $f(x)$.

Step3: Pack the matrices $M(0)$, and $M(j)$:

$$M(0) = \begin{bmatrix} 3 & a & s_2 \\ a & s_2 & s_3 \\ s_2 & s_3 & s_4 \end{bmatrix} \qquad M(j) = \begin{bmatrix} s_j & s_{j+1} & s_{j+2} \\ s_{j+1} & s_{j+2} & s_{j+3} \\ s_{j+2} & s_{j+3} & s_{j+4} \end{bmatrix}$$

compute the inner products of $\mathbf{s}_v$ and the first column of $M(0)^{-1}M(j)$, which gives s_{v+j}, (s_{j+3} and s_{j+4} are computed from the linear recursive relation from $\mathbf{s}_j$).

ElGamal-like Digital Signature Algorithm of Degree 3

- System public keys: p, a prime, $q = p^v$, Q, a prime factor of $q^2 + q + 1$ for $v \neq 2$ and $Q = P_1P_2$, $P_1 \mid p^2 + p + 1$, $P_2 \mid p^2 - p + 1$ for $v = 2$ res., and $f(x) = x^3 - a\,x^2 + bx - 1$, irreducible over GF($q$) with period Q
- $h(.)$: a hash function (SHA-1)
- Signer, private key: $0 < x < Q$ with $(x, Q) = 1$, public key $y = (s_x, s_{x+1}, x_{x+2})$.

Signer

- randomly picks k: $0 < k < Q$ coprime with Q (per message)
- applying Algorithm 2 to compute (s_k, s_{-k})
- setting r, an integer converted from s_k
- solves for t in the equation:
  $h(m) \equiv xr + kt \pmod{Q}$

(r, t) is a digital signature of the message m (s_{-k} needs to be transmitted)

Verifier

- setting $v = -h(m)t^{-1} \bmod Q$
  $u = -r\, t^{-1} \bmod Q$
- computes $A = s_{v+x}$ by Algorithm 3
- by Algorithm 2, computes
  $B = s_u(f_k)$, the uth term of the char. sequence of the LFSR

$$f_k(x) = x^3 - s_k x^2 + s_{-k} x - 1$$

- checks whether

$$A = B \qquad (1)$$

If (1) is true, accepts. Otherwise, rejects.

Signer: x

(s_x, s_{x+1}, s_{x+2}) (Alg 2)

Signing for the message m:

k

$f_k = (s_k, s_{-k})$ (Alg 2)

$r = \mathrm{convert}(s_k)$

x, k, m → $h(m) \equiv xr + kt \pmod{Q}$

t

Signature: (r, t).

Verifier:

m, r, $h(x)$

t, r

$v = -h(m)r^{-1}$

$u = -tr^{-1}$

(s_x, s_{x+1}, s_{x+2})

$A = s_{v+x}$ (Alg 3)

$B = s_u(f_k)$ (Alg 2)

(s_k, s_{-k})

$A = ?\, B$

Yes, accept it.

No, reject it.

Sign and Verifi. of Cubic DSA

- System public keys: $q = p = 5$, $Q = 5^2 + 5 + 1 = 31$, $f(x) = x^3 - (2x^2 + 1)$, irreducible over GF(5) with period 31 (in E.g. 1)
- $h(.) = 4x \pmod{31}$: a hash function.
- Signer, private key: $0 < x = 7 < 31$, public key $(s_7, s_8, s_9) = (2,4,4)$

Signer

to sign message $m = 10$

- randomly picks $k = 4$.
- by Algorithm 2, computes $(s_4, s_{-4}) = (2,0)$
- setting $r = s_k = 2$.
- solves for t:

$h(m) = 4m \equiv 9 \pmod{31}$

$$t \equiv k^{-1}(h(m) - xr)$$
$$= 4^{-1}(9 - 7\times 2) \equiv 22 \pmod{31}$$

(2, 22) is a digital signature of the message $m = 10$ ($s_{-k} = 0$ needs to be transmitted)

Verifier

- setting $v = -h(m)t^{-1} = -9 \times 16 \equiv 11 \pmod{31}$

$u = -r\,t^{-1} = -22 \times 16 \equiv 20 \pmod{31}$

- computes $A = s_{v+x}$ by Algorithm 3:

(1) Algorithm 2: $\mathbf{s}_v = \mathbf{s}_{11} = (0,1,3)$

(2)

$$M(7) = \begin{bmatrix} s_7 & s_8 & s_9 \\ s_8 & s_9 & s_{10} \\ s_9 & s_{10} & s_{11} \end{bmatrix} = \begin{bmatrix} 2 & 4 & 4 \\ 4 & 4 & 3 \\ 4 & 3 & 0 \end{bmatrix}$$

$$M(0) = \begin{bmatrix} 3 & 0 & 3 \\ 0 & 3 & 3 \\ 3 & 3 & 2 \end{bmatrix} \Rightarrow M(0)^{-1} = \begin{bmatrix} 3 & 1 & 4 \\ 1 & 3 & 4 \\ 4 & 4 & 1 \end{bmatrix}$$

$$M(0)^{-1}M(7) = \begin{bmatrix} 1 & 3 & 0 \\ 0 & 3 & 3 \\ 3 & 0 & 3 \end{bmatrix}$$

$A = s_{11+x} = \mathbf{s}_{11} \cdot (1,0,3) = 4$, the first column of $M(0)^{-1}M(7)$

- Algorithm 2: $B = s_{20}(f_4) = 4$, where $f_4(x) = x^3 - 2x^2 - 1$
- Since $A = B = 4$, accept it.

Sequences in the example of cubic DSA

$f(x) = x^3 + x - 1$, irreducible over $GF(5)$ with period 31. The characteristic sequence generated by $f(x)$:

$S =$

$3 \quad 0 \quad 3 \quad 3 \quad \boxed{2} \quad 0 \quad 1 \quad \boxed{2 \quad 4 \quad 4}$

$3 \quad \boxed{0 \quad 1 \quad 3} \quad 4 \quad 3 \quad 4 \quad 1 \quad \boxed{4} \quad 3$

$2 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad \boxed{0} \quad 4 \quad 1$

1

A

The 4-decimation of S ($k = 4$), which is generated by

$$f_4(x) = x^3 - 2x^2 - 1$$

is given by

$3 \quad 2 \quad 4 \quad 1 \quad 4 \quad 2 \quad 0 \quad 4 \quad 0 \quad 0$

$4 \quad 3 \quad 1 \quad 1 \quad 0 \quad 1 \quad 3 \quad 1 \quad 3 \quad 4$

$\boxed{4} \quad 1 \quad 1 \quad 1 \quad 3 \quad 2 \quad 0 \quad 3 \quad 3 \quad 1$

0

B

Red marked terms are computed by Signer, and the green ones computed by Verifier.

Profile of cubic DSA for the version $q = p$

- Security: the difficulty of solving discrete logarithm in the finite field $GF(p^3)$
- 341 bits Cubic DSA $\Leftrightarrow$ 170 bits EC-DSA
  $\Leftrightarrow$ 1024 bit RSA
  $\Leftrightarrow$ 1024 bits DSA

Related Public-key XTR (Lenstra and Verheul, 2000) ¾ using special characteristic sequences

System public parameters:

p, a prime number and $q = p^2$

$f(x) = x^3 - a x^2 + \boxed{a^p} x - 1$, irreducible over GF($q$) with period $Q \mid p^2 - p + 1$

Applications: The Basic Internet Key Exchange (IKE) Protocol

System setup: Primes p, q, $q|p$-1, and g of order q in $Z_p{}^*$. Each user has a private key for a signature algorithm SIG, and all have the public verification keys of the other users in the network. The protocol also uses a message authentication code: MAC, and a pseudorandom function generator PRF.

The protocol messages: A = Initiator , B = Responser

Start message (A $\rightarrow$ B): s, g^a

Response message (B $\rightarrow$ A): $s, ID_b, SIG_b(\text{"1"}, s, g^a, g^b), MAC_{k_1}(\text{"1"}, s, ID_b)$

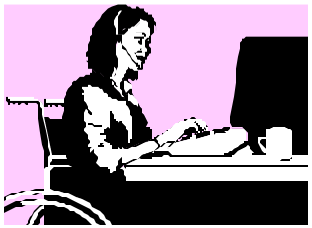
Finish message (A $\rightarrow$ B): $ID_a, SIG_a(\text{"0"}, s, g^b, g^a), MAC_{k_1}(\text{"0"}, s, ID_a)$

(Optional) ACK message (B $\rightarrow$ A): $MAC_{k_1}(\text{"1"})$

The protocol messages passing

Alice

Bob



s, g^a

$s, ID_b, SIG_b("1", s, g^a, g^b), MAC_{k_1}("1", s, ID_b)$

$ID_a, SIG_a("0", s, g^b, g^a), MAC_{k_1}("0", s, ID_a)$

$MAC_{k_1}("1")$

(IKE does not have this round currently!)

The shared session key: $k_0 = PRF_{g^{ab}}(0)$ MAC key: $k_1 = PRF_{g^{ab}}(1)$

The DH and DSA can be replaced by the cubic DH and cubic DSA.

Blackberry Screen Captures: System Setup.

GH-DH = cubic DH, GH-DSS = cubic DSA, are implemented at RIM's Blackberry handheld.

Fast Algorithm to Evaluate the *k*th Term of Char. Sequences of Degree 5

A. Characteristic Sequences of Degree 5

- Let $q = p^v$ and $f(x) = x^5 - ax^4 + bx^3 - cx^2 + dx - 1, \quad a,b,c,d \in GF(q)$ be irreducible over $GF(q)$ and a be a root of $f(x)$ in $GF(q^5)$.
- The characteristic sequence $\{s_k\}$ generated by $f(x)$ is given by

$$s_{k+5} = as_{k+4} - bs_{k+3} + cs_{k+2} - ds_{k+1} + s_k, \quad \mathrm{k} = 0,1,\cdots$$

with the initial state:

$$s_0 = 5,\ s_1 = a,\ s_2 = a^2 - 2b,\ s_3 = a^3 - 3ab + 3c,$$

$$s_4 = a^4 - 4a^2b + 2b^2 - 4d + 4ac$$

or equivalently,

$$s_k = Tr(a^k), \quad k = 0,1,\cdots$$

- The XTR analogue: let $q = p^2$, and the period of $f(x)$, say Q, is a factor of $p^4 - p^3 + p^2 - p + 1$, then

$$f(x) = x^5 - ax^4 + bx^3 - b^p x^2 + a^p x - 1, \quad a,b \in GF(p^2)$$

We may write the minimal polynomial of a^k as follows:

$$f_k(x) = x^5 - s_k x^4 + t_k x^3 - t_k^p x^2 + s_k^p x - 1$$

where $S = \{s_k\}$ and $\{s_{-k}\}$, $T = \{t_k\}$ and $\{t_{-k}\}$ are pairs of reciprocal sequences, and $s_{-k} = s_k^p$ and $t_{-k} = t_k^p$.

Lemma 1. *For all n, m,*

1. $s_{2n} = s_n^2 - 2t_n$
2. $t_{2n} = t_n^2 + 2s_n^p - 2s_n t_n^p$
3. $s_{3n} = s_n^3 - 3s_n t_n + 3t_n^p$
4. $t_{3n} = t_n^3 - 3s_n^p t_n - 3s_n t_n t_n^p + 3s_n^2 s_n^p + 3t_n^{2p} - 3s_n$
5. $s_{n+m} = s_n s_m - s_{n-m} t_m + s_{n-2m} t_m^p - s_{n-3m} s_m^p + s_{n-4m}$
6. $t_n t_m - s_m^p t_{n-m} + 3t_{n+m} = s_n s_m s_{n+m} - s_{n-2m} s_{n-m} + s_{2n-3m} - s_{n+2m} s_n - s_{2n+m} s_m + s_{n+m}^2$

Algorithm 4. Fifth-Order Algorithm for Evaluating the kth Terms of S and T Sequences.

1. Let $k = \sum_{i=0}^{n} c_i 3^i,\ c_i \in \{-1,0,1\}$.
2. Set $m = 1$ and $u = (s_{-1}, s_0, s_1, s_2, s_3)$ and $v = (t_{-1}, t_0, t_1, t_2, t_3)$
3. For $i = 0, \dots, n$
   - (a) Set $d_i = 3m + c_i$, and compute $u = (s_{d_i-2}, s_{d_i-1}, s_{d_i}, s_{d_i+1}, s_{d_i+2})$ and $v = (t_{d_i-2}, t_{d_i-1}, t_{d_i}, t_{d_i+1}, t_{d_i+2})$
   - (b) $m = d_i$

Output (s_m, t_m)

In the loop 3-(a), use the following relations, obtained from Lemma 1.

Term	Formula
s_{3m}	$s_{3m} = s_m(s_{2m} - t_m) + 3t_m^p$
s_{3m+1}	$s_{3m+1} = s_{2m}s_{m+1} - s_{m-1}t_{m+1} + s_2^p t_{m+1}^p - s_{m+3}^p s_{m+1}^p - s_{2m+4}$
s_{3m+2}	$s_{3m+2} = s_{2m+2}s_m - s_{m+2}t_m + s_2 t_m^p - s_{m-2}^p s_m^p - s_{2m-2}^p$
t_{3m}	$t_{3m} = t_m(t_{2m} + s_m^p) + s_m(3s_m s_m^p - t_m t_m^p - 9) + 3t_{2m}^p$
t_{3m+1}	$t_{3m+1} = [s_{m+1}(s_{2m}s_{3m+1} - s_{4m}s_{m+1} - s_{m-3}s_{m+1}^p + s_{3m-1}t_{m+1} - s_{2m-2}t_{m+1}^p - s_4^p)$ $+s_{m+1}^p t_{m-1} - s_2^p s_{m-1} + s_{m-3} - s_{4m+2}s_{2m} + s_{3m+1}^2 - t_{2m}t_{m+1}]/3$
t_{3m+2}	$t_{3m+2} = [s_{m+2}(s_{2m}s_{3m+2} - s_{4m+2}s_m - s_{m+2}s_m^p + s_{3m+2}t_m - s_{2m+2}t_m^p - s_2)$ $-s_{3m-2}^p s_{m-2}^p - s_{4m+4}s_{2m} + t_{m-2}^p s_{2m}^p + s_{4m-4}^p + s_{3m+2}^2 - t_{2m}t_{m+2}]/3$

Table 3. Sample Formulae for s and t Terms

Comparisons of Several Approaches for Computing the *k*th Terms of *S* and *T* Sequences and Representations

Algorithm	# Adds in $GF(p)$	# Mults in $GF(p)$	# of Bits
Exponentiation in $GF(p^{10})$	$178 \log l$	$52.5 \log l$	$10 \log p$
Root-Finding	$70925 \log p + 890 \log l$	$17400 \log p + 262.5 \log l$	$4 \log p$
Polynomial Extension	$890 \log l$	$262.5 \log l$	$4 \log p$
Fifth-Order	$280.1 \log l$	$108.5 \log l$	$4 \log p$

Table 4. Algorithmic Average Computational Cost and Bandwidth

XTR-Analogue of Quintic DSA

-System public keys: p, a prime, $q = p^2$, Q, a prime factor of $p^4 - p^3 + p^2 - p + 1$ for

$$f(x) = x^5 - ax^4 + bx^3 - b^p x^2 + a^p x - 1, \quad a, b \in GF(p^2)$$

irreducible over GF(q) with period Q

- $h(.)$: a hash function (SHA-1)
- Signer, private key: $0 < x < Q$ with $(x, Q) = 1$, public key: $\mathbf{s}_x = (s_x, s_{x+1}, s_{x+2}, s_{x+3}, s_{x+4})$

Signer

- randomly picks k: $0 < k < Q$ coprime with Q (per message)
- applying Algorithm 2 to compute

$$(s_k, t_k)$$

- setting r, an integer converted from s_k
- solves for t in the equation:

$$h(m) \equiv xr + kt \pmod{Q}$$

(r, t) is a digital signature of the message m (t_k needs to be transmitted)

Verifier

- setting $v = -h(m)t^{-1} \bmod Q$
  $u = -r\, t^{-1} \bmod Q$
- computes $A = s_{v+x}$ by Algorithm 1 (in which Assumption 1 is replaced by Algorithm 4)
- by Algorithm 4, computes $B = s_u(f_k)$, the uth term of the char. sequence of the LFSR

$$f_k(x) = x^5 - s_k x^4 + t_k x^3 - t_k^p x^2 + s_k^p x - 1$$

- checks whether

$$A = B \qquad (1)$$

If (1) is true, accepts. Otherwise, rejects.

The Contents of the talk is taking from the following research work:

1. G. Gong and L. Harn, A new approach for public key distribution, *the Proceedings of China-Crypto'98*, May 1998, Chengdu, China.
2. G. Gong and L. Harn, Public-key cryptosystems based on cubic finite field extensions, *IEEE Trans. on Inform. Theory*, vol. 45, No.7, November 1999, pp. 2601-2605.
3. G. Gong, L. Harn and H.P. Wu, The GH public-key cryptosystems, *Selected Areas in Cryptography*, *Lecture Notes in Computer Science* , S. Vaudenay and A. M. Youssef (Ed). Berlin, Germany, Springer-Verlag, 2001, vol. 2259, p.284-300.
4. K. Giuliani and G. Gong, Analogues to the Gong-Harn and XTR cryptosystems, Technical report, University of Waterloo, CORR 2003-34, accessible at www.cacr.math.uwaterloo.ca.
5. K. Giuliani and G. Gong, Efficient key agreement and signature schemes using compact represenations in $GF(p^{10})$, will be appeared at the Proceedings of the ISIT 2004, July 2004.
6. K. Giuliani and G. Gong, Signature schemes based on the trace discrete log problem (Trace-DLP), will be appeared soon as Technical report, University of Waterloo, February, 2004 (submitted to Crypto04).

References of Some Related Work

- W. Diffie and M.E. Hellman, “New directions in cryptography,” *IEEE Trans. On Inform. Theory*, vol. IT-22, November 1976, pp.644-654.

  Comments: Exponentiation in DH can be considered as evaluating k^{th} term of a first order LFSR sequence over GF(q).

- W.B. Müller and W. Nöbauer, "Cryptanalysis of the Dickson-scheme, " *Advances in Cryptology*, *Proceedings of Eurocrypt'85*, pp. 71-76.
- P. Smith, "LUC public-key encryption, " *Dr. Dobb's Journal*, pp. 44-49, January 1993.

  Comments: The mathematical function used in this family of the public-key cryptosystems is a 2^{nd}-order LFSR characteristic sequence over GF(p).

- A.K. Lenstra and E.R. Verheul, The XTR public key systems, *Advances in Cryptology*, *Proceedings of Crypto2000*, pp. 1-19, August, 2000.

  Comments: the mathematical function is a 3^{rd}-order LFSR characteristic sequence over GF(p^2) which is a special case of the sequences used in the GH public key cryptosystem.

- Karl Rubin and Alice Silverberg, *Torus-based cryptography*, *Advances in Cryptology*, *Proceedings of Crypto2003*, August 2003.

  Comments: Generalize GH and XTR in a general model using an algebraic tool: Tori.