An Implementation of Public Key Cryptosystem TTM with Linear Time Complexity for Decryption

Yuh-Hua Hu[1], Lih-Chung Wang[2], Jiun-Ming Chen[3], Feipei Lai[4], Chun-Yen Chou[5]

Abstract — **T. Moh invented a cryptosystem called tame transformation method (TTM). TTM cryptosystem is claimed to be the fastest among all currently known unbroken public key cryptosystems. We have done an actual implementation with linear time complexity for decryption and made some performance tests on it to verify the claim of speed. Also, we give some discussion on our TTM implementation and some attacks on TTM.**

I. Introduction

TTM [1] applies abstract algebra to cryptography, while the most well-known, such as RSA, ElGamal, use number theory. Although other recent public key cryptosystems, such as ECC, HFE, NTRU, NICE, don't use only elementary number theory any more. TTM is claimed to be the fastest, especially in decrypting. Since even for a naive implementation of TTM the time complexity of encryption per block is $O(mn^2)$ and decryption is $O(m^2)$, where n is the number of bytes of one plaintext block, and m is the number of bytes of one ciphertext block. We have done an actual implementation improving the decryption speed significantly.

II. TTM Cryptosystem

Let $\mathbb{F}$ denote a field and $n \in \mathbb{N}$. $\varphi : \mathbb{F}^n \longrightarrow \mathbb{F}^n$ is a *tame transformation* if φ is an invertible affine transformation, or, after a permutation of indices if necessary,

$$\varphi : \begin{cases} y_1 &= x_1 \\ y_2 &= x_2 + f_2(x_1) \\ &\vdots \\ y_n &= x_n + f_n(x_1, x_2, \cdots, x_{n-1}) \end{cases}$$

where each f_i is a polynomial, $i = 2, 3, \cdots, n$. Note that an affine transformation can be identified as applying an invertible matrix and adding a constant vector.

Given a plaintext message P, we apply an imbedding ρ from $\mathbb{F}^n$ to $\mathbb{F}^m$, then apply tame transformations $\varphi_1, \varphi_2, \varphi_3, \varphi_4$ subsequently to get X, Y, Z, C in $\mathbb{F}^m$ where C is the result cipher text. φ_1, φ_4 are invertible affine transformations, φ_2, φ_3 are nonlinear tame transformations. In diagram,

$$\begin{array}{ccccccccc} & & \varphi_1\rho & & \varphi_2 & & \varphi_3 & & \varphi_4 & \\ \mathbb{F}^n & \longrightarrow & \mathbb{F}^m & \longrightarrow & \mathbb{F}^m & \longrightarrow & \mathbb{F}^m & \longrightarrow & \mathbb{F}^m \\ P & \longmapsto & X & \longmapsto & Y & \longmapsto & Z & \longmapsto & C. \end{array}$$

III. Implementation of TTM

1. φ_1, φ_4 can be identified as applying an invertible matrix and adding a constant vector. However, we can generate any invertible matrix with elementary row operations. We use the type of elementary matrices $E_{(i,j,\lambda)}$: first multiply the i-th row by a scale λ, then add it to j-th row. The number of elementary row operations at φ_1 and φ_4 are $2n + R$ and $2m + R$ where R is a constant.
2. φ_2 and φ_3 are the most important parts in TTM. We use two different schemes to create them.

IV. Decryption Time Complexity Analysis

The decryption consists of applying, in the following order, the four pointwise-inverses $\varphi_4^{-1}, \varphi_3^{-1}, \varphi_2^{-1}, \varphi_1^{-1}$.

1. φ_1^{-1} is done by applying the corresponding elementary row operations for φ_1 but in the reverse order. The time complexity is $O(n)$ per block. Similarly, φ_4^{-1} is $O(m)$.
2. φ_2^{-1} is m quadratic polynomials with n variables. The number of the monomials for each polynomial in φ_2 is less than a constant, say M_2. Each monomial needs at most two multiplications. Thus each polynomial needs at most $2M_2$ multiplications and $M_2 - 1$ additions. The total time complexity is $O(mM_2) = O(m)$ per block. φ_3^{-1} is even simpler. So the time complexity is $O(m)$.

For one block, the time complexity of decryption is $O(m)$. For one byte, the time complexity of decryption is $O(\frac{m}{n})$. This means that we can increase the security of TTM with large n and m and still keep its fast decryption.

V. Discussion

1. If we limit the coefficients of the monomials in public key and private key in subfield of $GF(2^4)$, then we can reduce the key size approximately by half.
2. One concern is about the data expansion rate $\frac{m}{n}$. Due to design requirement for φ_3, we must have $m > n$. But $m - n$ is actually less than a constant. Therefore, we can reduce the data expansion rate by using large m, n.
3. The private key of our implementation is weaker than the original design, but without even bother to count the number of choices for φ_2 and φ_3, there are about 2^{2399} different private key choices. Therefore it is infeasible to use the exhaustive search for the private key.
4. There are some attacks,such as Isomorphism of Polynomials, Solving Nonlinear Equations, MinRank Problem. But they are infeasible.

References

[1] T. Moh, "A Public Key System With Signature And Master Key Functions", Communications in Algebra, 27(5), 2207-2222 (1999).

[1]Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan e-mail: r90010@csie.ntu.edu.tw

[2]Department of Applied Mathematics, National Donghwa University, Hualien 974, Taiwan e-mail: lcwang@mail.ndhu.edu.tw

[3]Department of Mathematics, Purdue University, West Lafayette, IN 47907, USA e-mail: jmchen@math.purdue.edu

[4]Department of Electrical Engineering & Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan e-mail: flai@cc.ee.ntu.edu.tw

[5]Corresponding Author, Department of Mathematical Education, National Hualien Teachers College, Hualien 970, Taiwan e-mail: choucy@sparc2.nhltc.edu.tw

0-7803-7728-1/03/$17.00