PQCrypto 2006

International Workshop on Post-Quantum Cryptography

23–26 May 2006

Leuven, Belgium

Program committee:

Daniel J. Bernstein, University of Illinois at Chicago, USA
Johannes Buchmann, Technische Universität Darmstadt, Germany
Jintai Ding, University of Cincinnati, USA
Louis Goubin, Université de Versailles, France
Tanja Lange, Danmarks Tekniske Universitet, Denmark
Phong Nguyen, École Normale Supérieure, France
Tatsuaki Okamoto, NTT Laboratories, Japan
Louis Salvail, Aarhus Universitet, Denmark
Alice Silverberg, University of California at Irvine, USA
Joe Silverman, Brown University, USA
Martijn Stam, École Polytechnique Fédérale de Lausanne, Switzerland
Christopher Wolf, École Normale Supérieure, France

Invited speakers:

Sean Hallgren, NEC Laboratories, USA
Phong Nguyen, École Normale Supérieure, France
Oded Regev, Tel-Aviv University, Israel
Nicolas Sendrier, Institut National de Recherche en Informatique et en Automatique, France
Jacques Stern, École Normale Supérieure, France
Michael Szydlo, RSA Laboratories, USA
Lieven Vandersypen, Technische Universiteit Delft, the Netherlands

Contributors:

Koichiro Akiyama, Toshiba Corporation, Japan
Chia-Hsin Owen Chen, National Taiwan University, Taiwan
Jiun-Ming Chen, National Taiwan University / Chinese Data Security, Inc., Taiwan
Jintai Ding, University of Cincinnati, USA
Jean-Charles Faugère, LIP6, France
Ryou Fujita, Institute of Information Security, Japan
Nicolas Gama, École Normale Supérieure, France
Yasuhiro Goto, Hokkaido University of Education, Japan
Aline Gouget, Gemplus, France
Jason Gower, University of Cincinnati, USA
Jeff Hoffstein, Brown University / NTRU Cryptosystems, USA
Nick Howgrave-Graham, NTRU Cryptosystems, USA
Lei Hu, Chinese Academy of Sciences, China
Atsushi Kanai, NTT Information Sharing Platform Laboratories, Japan
Elham Kashefi, Institute for Quantum Computing, Canada
Iordanis Kerenidis, CNRS, Universite Paris-Sud, France
Tetsutaro Kobayashi, NTT Information Sharing Platform Laboratories, Japan
Jianyu Li, Chinese Academy of Sciences, China
Toshiyuki Miyazawa, NTT Information Sharing Platform Laboratories, Japan
Ichizo Nakamura, NTT Information Sharing Platform Laboratories, Japan
Phong Nguyen, École Normale Supérieure, France
Xuyun Nie, Chinese Academy of Sciences, China
Satoshi Oda, NTT Information Sharing Platform Laboratories, Japan
Jacques Patarin, University of Versailles, France
Ludovic Perret, Université Catholique de Louvain, Belgium
Jill Pipher, Brown University / NTRU Cryptosystems, USA
Bart Preneel, Katholieke Universiteit Leuven, Belgium
Dieter Schmidt, University of Cincinnati, USA
Joseph H. Silverman, Brown University / NTRU Cryptosystems, USA
Kohtaro Tadaki, Chuo University, Japan
Shigeo Tsujii, Institute of Information Security, Japan
John Wagner, University of Cincinnati, USA
William Whyte, NTRU Cryptosystems, USA / Ireland
Christopher Wolf, École Normale Supérieure, France
Bo-Yin Yang, Tamkang University, Taiwan

Local organization:

Nessim Kisserli
Özgül Küçük
Joseph Lano
Péla Noë
Jasper Scholten
Koen Simoens
Saartje Verheyen
Christopher Wolf
Elvira Wouters

Program and table of contents:

Can quantum computers be built?

Lieven Vandersypen

Technische Universiteit Delft

Quantum algorithms

Sean Hallgren

NEC Laboratories

Post-quantum hash-based cryptography

Michael Szydlo

RSA Laboratories

The Merkle-tree construction promises to offer authentication and digital signatures which are resistant to quantum attacks. One of the first proposals in public key cryptography (1979), this construction relies only on a hash function for its security. While RSA and other number theory based algorithms will succumb to efficient quantum algorithms, a hash function need not have a "number-theoretic" basis. Without this structure, a quantum Fourier transform based algorithm seems less likely to appear. In this talk we will review the Merkle tree constructions and focus on recent advances which make the Merkle-tree construction more efficient and practical. We explain the space-time tradeoffs inherent in the Merkle construction, which historically made them less appealing than other authentication and digital signature schemes. However, we suggest that recent efficiency improvements render Merkle trees more practical today, and the threat of quantum algorithms makes this well accepted construction even more appealing.

Statistical Zero Knowledge and quantum one-way functions

Elham Kashefi
IQC - University of Waterloo &
Christ Church - University of Oxford
ekashefi@iqc.ca

Iordanis Kerenidis
Dept. of Mathematics
MIT
jkeren@math.mit.edu

May 2, 2006

Abstract

One-way functions are a fundamental notion in cryptography, since they are the necessary condition for the existence of secure encryption schemes. Most examples of such functions, including factoring, discrete Logarithm or the RSA function, can be, however, inverted with the help of a quantum computer. Hence, it is very important to study the possibility of *quantum one-way functions*, i.e. functions which are easily computable by a classical algorithm but are hard to invert even by a quantum adversary. In this paper, we provide for the first time a set of problems that are good candidates for quantum one-way functions. These problems include Graph Non-Isomorphism, approximate Closest Lattice Vector and Group Non-Membership. More generally, we show that any hard instance of Circuit Quantum Sampling gives rise to a quantum one-way function. By the work of Aharonov and Ta-Shma [2], this implies that any language in Statistical Zero Knowledge which is hard-on-average for quantum computers, leads to a quantum one-way function. Moreover, extending the result of Impagliazzo and Luby [9] to the quantum setting, we prove that quantum distributionally one-way functions are equivalent to quantum one-way functions.

1 Introduction

One-way functions are at the core of modern cryptography. The fundamental task of cryptography is that of secure encryption of information against malicious parties. The existence of such secure encryption schemes implies that there is an efficient way of generating instances of problems together with some auxiliary information, such that it is easy to solve these instances with the help of the auxiliary information but hard to solve on average without it.

This concept is exactly captured by the definition of one-way functions, which are the necessary condition for the existence of cryptography. Moreover, one-way functions have many theoretical applications, for example in their connections to cryptographic primitives like bit commitment and oblivious transfer, Zero Knowledge Proof Systems and pseudorandom generators.

However, proving that one-way functions exist would imply that P $\neq$ NP and hence, we only have "candidate" one-way functions. Such candidate problems include Factoring, Discrete Logarithm, Graph Isomorphism, Quadratic Residuosity, approximate Shortest Vector and Closest Vector and the RSA function. These problems seem to belong to a class called NP-*Intermediate*, i.e. they are NP problems for which we do not know any efficient algorithm, but they don't seem to be NP-hard. Moreover, many of the candidate problems belong to the class of Statistical Zero Knowledge (SZK). In fact, Ostrovsky [15] showed that if SZK contains any *hard-on-average* problem, then one-way functions exist.

The emergence of quantum computation and communication has provided the field of cryptography with many new strengths and challenges. The possibility of unconditionally secure key distribution shows that the laws of quantum mechanics can allow for the secure transmission of information over quantum channels. Moreover, Shor's celebrated algorithm for factoring and discrete logarithm implies that many classical one-way functions and hence cryptosystems, including RSA, will not be secure against quantum adversaries. It is a very important question to ask whether we can construct cryptosystems which are secure even against quantum attacks. To this end, we need to find good candidates for quantum one-way functions, i.e. functions which are easily computable by a classical algorithm but hard to invert even by a quantum adversary.

Several other applications of quantum one-way functions have also been studied in a series of papers. For example, the connections between quantum one-way functions and quantum computationally secure bit commitment schemes were explored in [4, 1, 3]. On the other hand, Gottesman *et.al.* [6] proposed a digital signature scheme based on a quantum one-way function with classical inputs but quantum outputs and proved the informational security of their protocol. Moreover, Kashefi *et.al.* [10] and Kawachi *et.al.* [11] presented a necessary and sufficient condition for testing the one-wayness of a given permutation in the quantum setting based on the efficiency of constructing a family of reflection operators. Recently, Watrous [18] proved that several classical interactive proof systems are statistically zero-knowledge against quantum attacks. In addition, he showed that Computational Zero Knowledge against quantum attacks for NP is implied by the existence of quantum one-way permutations.

Despite the importance of the applications of quantum one-way functions, there had been no results so far that provided good candidate problems. Here, we prove the quantum analogue of Ostrovsky's result and show that if there exists a problem in Statistical Zero Knowledge which is hard-on-average for a quantum computer, then quantum one-way functions exist. This is the first result that provides a set of problems that are good candidates for quantum one-way functions.

The key insight in our result is the connection of quantum one-way functions to the problem of *Circuit Quantum Sampling*. Informally speaking, *quantum sampling* is the ability to prepare efficiently a superposition that corresponds to a samplable classical probability distributions, i.e. a superposition whose amplitudes are the square roots of the probabilities of a classical distribution from which one can efficiently sample. The hardness of this task depends on the structure of the underlying set. For example, it is well known that being able to quantumly sample from the set of homomorphisms of a given input graph is sufficient to solve the notorious Graph Isomorphism problem. Aharanov and Ta-shma [2] have introduced this framework of circuit quantum sampling and have shown that many problems in quantum computation, including Graph Isomorphism, Discrete Logarithm, Quadratic Residuosity and approximate Closest Lattice Vector (CVP), are all instances of it.

We relate the problem of quantum sampling to quantum one-way functions by giving a simple proof that any hard instance of the quantum sampling problem implies the existence of a quantum one-way function. We first prove our results for the case of one-to-one one-way functions, the existence of which seems to be a stronger assumption than that of general one-way functions. Then, we generalize our results for many-to-one one-way functions. We show that a hard instance of the CQS problem implies a quantum distributionally one-way function and then prove that a quantum distributionally one-way function implies a quantum one-way function. The notion of classical distributionally one-way functions was introduced by Impagliazzo-Luby in [9], where they also prove their equivalence to classical one-way functions.

Aharonov and Ta-Shma showed that any Statistical Zero Knowledge language (SZK) can be reduced to a family of instances of the CQS problem. Using our result that a hard instance of CQS

implies the existence of a quantum one-way function, we conclude that if there exists a language in Statistical Zero Knowledge which is hard-on-average, then quantum one-way functions exist.

2 Preliminaries

In this section we provide a brief overview of classical one-way functions and quantum computation. For an excellent exposition on quantum computation we refer the reader to [14] and for one-way functions to [5].

2.1 Classical one-way functions

Definition 1 *A function $f : \{0,1\}^* \to \{0,1\}^*$ is a* one-way function*, if the following conditions are satisfied:*

(i) easy to compute: *f can be computed by a polynomial size classical circuit.*

(ii) hard to invert: *There exists a polynomial $p(\cdot)$ such that for any probabilistic polynomial time algorithm I and for all sufficiently large $n \in \mathbf{N}$,*

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} \text{Prob}[I(f(x), 1^n) \in f^{-1}(f(x))] \leq 1 - \frac{1}{p(n)}$$

A classical one-way function f is defined in terms of uniform family of functions f_n, one for each input length n. The inverter I of the function takes as input the value $f(x)$ and the size n in unary. For simplicity, in the following definitions we omit the parameter n. One can also assume, without loss of generality that the function f, is *length regular i.e.* for every $x, y \in \{0,1\}^*$, if $|x| = |y|$ then $|f(x)| = |f(y)|$ and *length preserving i.e.* for every $x \in \{0,1\}^*$, $|f(x)| = |x|$ (see [5]).

Furthermore, Impagliazzo and Luby [9] defined a seemingly weaker notion of one-wayness for many-to-one functions, called *distributionally one-way function*, and proved that, in fact, the existence of a distributionally one-way function implies the existence of a one-way function.

Definition 2 *A function $f : \{0,1\}^* \to \{0,1\}^*$ is a* distributionally one-way function*, if the following conditions are satisfied:*

(i) easy to compute: *f can be computed by a polynomial size classical circuit.*

(ii) hard to sample: *There exists a polynomial $p(\cdot)$ such that for any probabilistic polynomial time algorithm S and for all sufficiently large $n \in \mathbf{N}$, the distribution defined by $(x, f(x))$ and the distribution defined by $(S(f(x)), f(x))$ are statistically distinguishable by at least $\frac{1}{p(n)}$ when $x \in \{0,1\}^n$ is chosen uniformly.*

2.2 Quantum Computation

Let H denote a 2-dimensional complex vector space, equipped with the standard inner product. We pick an orthonormal basis for this space, label the two basis vectors $|0\rangle$ and $|1\rangle$, and for simplicity identify them with the vectors $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$, respectively. A *qubit* is a unit length vector in this space, and so can be expressed as a linear combination of the basis states:

$$\alpha_0|0\rangle + \alpha_1|1\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}.$$

Here α_0, α_1 are complex *amplitudes*, and $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

An *m-qubit system* is a unit vector in the m-fold tensor space $H \otimes \cdots \otimes H$. The 2^m basis states of this space are the m-fold tensor products of the states $|0\rangle$ and $|1\rangle$. For example, the basis states of a 2-qubit system are the four 4-dimensional unit vectors $|0\rangle \otimes |0\rangle$, $|0\rangle \otimes |1\rangle$, $|1\rangle \otimes |0\rangle$, and $|1\rangle \otimes |1\rangle$. We abbreviate, *e.g.* , $|1\rangle \otimes |0\rangle$ to $|0\rangle|1\rangle$, or $|1, 0\rangle$, or $|10\rangle$, or even $|2\rangle$ (since 2 is 10 in binary). With these basis states, an m-qubit state $|\phi\rangle$ is a 2^m-dimensional complex unit vector

$$|\phi\rangle = \sum_{i \in \{0,1\}^m} \alpha_i |i\rangle.$$

We use $\langle\phi| = |\phi\rangle^*$ to denote the conjugate transpose of the vector $|\phi\rangle$, and $(\phi\ ,\ \psi) = \langle\phi| \cdot |\psi\rangle$ for the inner product between states $|\phi\rangle$ and $|\psi\rangle$. These two states are *orthogonal* if $(\phi\ ,\ \psi) = 0$. The *norm* of $|\phi\rangle$ is $\|\phi\| = \sqrt{|(\phi\ ,\ \phi)|}$.

A quantum state can evolve by a unitary operation or by a measurement. A *unitary* transformation is a linear mapping that preserves the ℓ_2 norm. If we apply a unitary U to a state $|\phi\rangle$, it evolves to $U|\phi\rangle$.

The most general measurement allowed by quantum mechanics is specified by a family of positive semidefinite operators $E_i = M_i^* M_i$, $1 \leq i \leq k$, subject to the condition that $\sum_i E_i = I$. A projective measurement is defined in the special case where the operators are projections. Let $|\phi\rangle$ be an m-qubit state and $B = \{|b_1\rangle, \ldots, |b_{2^m}\rangle\}$ an orthonormal basis of the m-qubit space. A projective measurement of the state $|\phi\rangle$ in the B basis means that we apply the projection operators $P_i = |b_i\rangle\langle b_i|$ to $|\phi\rangle$. The resulting quantum state is $|b_i\rangle$ with probability $p_i = |(\phi\ ,\ b_i)|^2$.

2.3 Quantum Sampling

Let $\{C_i\}$ be a uniform classical circuit family and for every input size n define D_{C_n} to be the distribution over outputs of the circuit $C_n : \{0,1\}^n \to \{0,1\}^m$ when the input distribution is uniform. Denote by $|C_n\rangle = \sum_{z \in \{0,1\}^m} \sqrt{D_{C_n}(z)}|z\rangle$, the quantum sample of outputs of C_n.

Definition 3 *Given a uniform family of classical circuit $\{C_i\}$ and a real number $0 \leq \epsilon < \frac{1}{2}$, define QS_C to be an efficient quantum circuit which for any sufficiently large input size n, prepares a state that is ϵ-close to the quantum sample $|C_n\rangle$, i.e. $|(QS_C(|0\rangle, 1^n)\ ,\ |C_n\rangle)|^2 \geq 1 - \epsilon$.*

The problem of finding such a quantum circuit QS_C for any given uniform family of classical circuits $\{C_i\}$ was introduced by Aharanov and Ta-shma in [2], as the *Circuit Quantum Sampling Problem* (CQS). In fact, they defined CQS as $|QS_C(|0\rangle, 1^n) - |C_n\rangle| \leq \epsilon$, however both definitions suffice for the proof that Statistical Zero Knowledge reduces to a family of instances of the CQS problem. We say that the quantum sampling problem for $\{C_i\}$ is *hard* if there exists no efficient QS for any constant $\epsilon \in [0, 1/2]$.

3 Definitions of quantum one-way functions

A *quantum one-way function* is defined similarly to the classical case, where now the inverter I is a polynomial size uniform quantum circuit family. For clarity, we follow again the convention of omiting the parameter of the input size n.

Definition 4 *A one-to-one function $f : \{0,1\}^* \to \{0,1\}^*$ is a* quantum one-way function*, if the following conditions are satisfied:*

(i) easy to compute: *f can be computed by a polynomial size classical circuit.*

(ii) hard to invert: *There exists a polynomial $p(\cdot)$ such that for any quantum polynomial time algorithm I and all sufficiently large $n \in \mathbf{N}$,*

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} \text{Prob}[I(f(x)) \in f^{-1}(f(x))] \leq 1 - \frac{1}{p(n)}$$

In the quantum case, the probability of success of the inverter I is defined as the square of the inner product between the outcome of I and the outcome of the perfect inverter P, where

$$P : |f(x)\rangle|\beta\rangle \mapsto |f(x)\rangle|x \oplus \beta\rangle$$

In other words, for the case of one-to-one functions

$$\text{Prob}[I(f(x)) \in f^{-1}(f(x))] = \text{Prob}[I(f(x)) = x] = |(I(|f(x)\rangle|\beta\rangle)\ ,\ |f(x)\rangle|x \oplus \beta\rangle)|^2.$$

One can also define another type of average case quantum one-way function (called strong one-way function), where we require that any quantum algorithm inverts the function with negligible probability (instead of just failing with non-negligible probability). However, similar to the classical case, if there exists a weak quantum one-way function (Definition 4), then there exists a strong quantum one-way function as well [7, 5, 10].

We now provide an alternative definition for a one-to-one quantum one-way function, which is more suitable for constructing the relation between quantum one-way functions and the CQS problems and prove the equivalence of the two definitions.

Definition 5 *A one-to-one function $f : \{0,1\}^* \to \{0,1\}^*$ is a* quantum one-way function *if:*

(i) f can be computed by a polynomial size classical circuit.

(ii) There exists a polynomial $p(\cdot)$ such that there exists no quantum polynomial time algorithm I' with the property that for all sufficiently large $n \in \mathbf{N}$:

$$I' : |f(x)\rangle|\beta\rangle \mapsto a_{f(x)}|f(x)\rangle|x \oplus \beta\rangle + b_{f(x)}|f(x)\rangle|G_{f(x)}\rangle \tag{1}$$

where $G_{f(x)}$ is a garbage state, $\frac{1}{2^n}\sum_{x\in\{0,1\}^n} a^2_{f(x)} \geq 1 - \frac{1}{p(n)}$ and $a_{f(x)}$ are positive real numbers.

It is clear that definition 4 implies definition 5 and we also prove the converse:

Theorem 1 *If a one-to-one function f is quantum one-way according to definition 5, then it is also quantum one-way according to definition 4.*

Proof. Let $f : \{0,1\}^* \to \{0,1\}^*$ be a quantum one-way function according to definition 5. Assume for contradiction that this function is not one-way according to definition 4. Then, for all polynomials $p(\cdot)$ there exists a quantum polynomial time algorithm I' with the property that for all sufficiently large $n \in \mathbf{N}$:

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} \text{Prob}[I(f(x)) \in f^{-1}(f(x))] \geq 1 - \frac{1}{p(n)}$$

or equivalently

$$I : |f(x)\rangle|\beta\rangle \mapsto c_{f(x)}|f(x)\rangle|x \oplus \beta\rangle + d_{f(x)}|\psi_{f(x)}\rangle \tag{2}$$

where $|\psi_{f(x)}\rangle$ is a garbage state and $\frac{1}{2^n}\sum_{x\in\{0,1\}^n}|c_{f(x)}|^2 \geq 1 - \frac{1}{p(n)}$. Without loss of generality we can assume that $c_{f(x)}$ are real numbers. We use this inverter to construct the following unitary that achieves the positive amplitudes. For clarity, here and in subsequent places in the paper we only show the unitary construction for the case where the ancilla registers are set to $|0\rangle$, unless the general ancilla state is required for the construction. It is clear of course how to unitarily extend the $|0\rangle$ ancilla to the other basis states.

$$\begin{array}{lll}
|f(x)\rangle|0\rangle|0\rangle|0\rangle & \rightarrow_{(\mathrm{CNOT})_{1,3}} & |f(x)\rangle|0\rangle|f(x)\rangle|0\rangle \\
& \rightarrow_{I_{1,2}} & (c_{f(x)}|f(x)\rangle|x\rangle + d_{f(x)}|\psi_{f(x)}\rangle)|f(x)\rangle|0\rangle \\
& \rightarrow_{I_{3,4}} & c^2_{f(x)}|f(x)\rangle|x\rangle|f(x)\rangle|x\rangle + c_{f(x)}d_{f(x)}|f(x)\rangle|x\rangle|\psi_{f(x)}\rangle + \\
& & d_{f(x)}c_{f(x)}|\psi_{f(x)}\rangle|f(x)\rangle|x\rangle + d^2_{f(x)}|\psi_{f(x)}\rangle|\psi_{f(x)}\rangle \\
& \rightarrow_{(\mathrm{CNOT})_{1,3}(\mathrm{CNOT})_{2,4}} & c^2_{f(x)}|f(x)\rangle|x\rangle|0\rangle|0\rangle + b_{f(x)}|\psi'_{f(x)}\rangle
\end{array}$$

where $|\psi'_{f(x)}\rangle$ is the new garbage state, orthogonal to the ideal state $|f(x)\rangle|x\rangle|0\rangle|0\rangle$ and by the fact that the average of the squares is larger than the square of the average we have

$$\tfrac{1}{2^n}\textstyle\sum_{x\in\{0,1\}^n} c^4_{f(x)} \geq (\tfrac{1}{2^n}\sum_{x\in\{0,1\}^n} c^2_{f(x)})^2 \geq (1 - \frac{1}{p(n)})^2 \geq 1 - \frac{1}{p'(n)}$$

Hence we have a new inverter

$$I' : |f(x)\rangle|\beta\rangle \mapsto a_{f(x)}|f(x)\rangle|x \oplus \beta\rangle + b_{f(x)}|\psi'_{f(x)}\rangle \tag{3}$$

with $\frac{1}{2^n}\sum_{x\in\{0,1\}^n} a^2_{f(x)} \geq 1 - \frac{1}{p(n)}$ and $a_{f(x)} = c^2_{f(x)}$ being positive real numbers. Finally, we can obtain the required form of the garbage state:

$$\begin{array}{lll}
|f(x)\rangle|0\rangle|0\rangle & \rightarrow_{(\mathrm{CNOT})_{1,2}} & |f(x)\rangle|f(x)\rangle|0\rangle \\
& \rightarrow_{I'_{2,3}} & a_{f(x)}|f(x)\rangle|f(x)\rangle|x\rangle + b_{f(x)}|f(x)\rangle|\psi'_{f(x)}\rangle \\
& \rightarrow_{(\mathrm{CNOT})_{1,2}} & a_{f(x)}|f(x)\rangle|0\rangle|x\rangle + b_{f(x)}|f(x)\rangle|G_{f(x)}\rangle
\end{array}$$

We reached a contradiction and therefore the function f is one-way according to definition 4. Note that for simplicity of presentation we dropped the $|0\rangle$ registers that are constant for all x. □

In the standard definition, a many-to-one function is called one-way if there exists no inverter that outputs with high probability an arbitrary preimage of $f(x)$. For many-to-one functions, Impagliazzo-Luby [9] defined a seemingly weaker notion, the *distributionally one-way function*. In this case, an inverter is required to output a *random* preimage of $f(x)$ and not just an arbitrary one. However, they prove that, in fact, the existence of a distributionally one-way function implies the existence of a one-way function. We also define quantum distributionally one-wayness for many-to-one functions and will prove its equivalence to the quantum one-way functions.

Definition 6 *A many-to-one function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a* quantum distributionally one-way function*, if the following conditions are satisfied:*

(i) f can be computed by a polynomial size classical circuit.

(ii) hard to invert: *There exists a polynomial $p(\cdot)$ such that for any quantum polynomial time algorithm S and all sufficiently large $n \in \mathbf{N}$,*

$$\frac{1}{2^n} \sum_{x \in \{0,1\}^n} |(S(|f(x)\rangle|0\rangle) \ , \ |f(x)\rangle|H_{f(x)}\rangle)|^2 \leq 1 - \frac{1}{p(n)},$$

where $|H_{f(x)}\rangle = \frac{1}{\sqrt{|f^{-1}(f(x))|}} \sum_{x \in f^{-1}(f(x))} |x\rangle$.

Note that one could potentially consider different definitions for quantum distributionally one-way functions, for example the quantum inverter could return a superposition with equal amplitudes but different phases. We believe that our quantum definition captures the essence of the classical one and moreover, we only use the above notion as an intermediate step in our proofs. Similar to the case of one-to-one functions we also give an equivalent definition

Definition 7 *A many-to-one function $f : \{0,1\}^* \to \{0,1\}^*$ is a* quantum distributionally one-way function *if:*

(i) f can be computed by a polynomial size classical circuit.

(ii) There exists a polynomial p such that there exists no quantum polynomial time algorithm S' with the property for all sufficiently large $n \in \mathbf{N}$

$$S' : |f(x)\rangle|0\rangle \mapsto a_{f(x)}|f(x)\rangle|H_{f(x)}\rangle + b_{f(x)}|f(x)\rangle|G_{f(x)}\rangle \tag{4}$$

where $|G_{f(x)}\rangle$ is a garbage state, $\frac{1}{2^n} \sum_{x \in \{0,1\}^n} a^2_{f(x)} \geq 1 - \frac{1}{p(n)}$, $a_{f(x)}$ are positive real numbers and $|H_{f(x)}\rangle = \frac{1}{\sqrt{|f^{-1}(f(x))|}} \sum_{x \in f^{-1}(f(x))} |x\rangle$.

We can easily extend the above algorithm S' into a unitary operation by mapping every other basis state $|f(x)\rangle|\beta\rangle$ to $|f(x)\rangle|T^{\beta}_{f(x)}\rangle$, where the set $\{|H_{f(x)}\rangle, T^1_{f(x)}, \ldots, T^{2^n-1}_{f(x)}\}$ is any orthonormal basis. Following the same steps as in the proof of Theorem 1 we have that

Theorem 2 *If a many-to-one function f is quantum one-way according to definition 7, then it is also quantum one-way according to definition 6.*

4 Circuit quantum sampling and one-way functions

In this section, we show that hard instances of the Circuit Quantum Sampling problem are good candidates for quantum one-way functions.

4.1 One-to-one one-way functions

We first focus our attention to the case of one-to-one one-way functions. The existence of one-to-one one-way functions is a seemingly stronger assumption than that of the existence of general one-way functions, since a one-way function doesn't immediately imply a one-to-one one-way function. However, this case illustrates the main ideas of our construction. In the following sections, we generalize our results for the case of many-to-one functions.

Theorem 3 *Assume for a classical circuit family $\{C_n\}$, which computes a one-to-one function, the corresponding CQS problem is hard , i.e. there exists no efficient quantum circuit implementing QS_C. Then the function $f : \{0,1\}^* \to \{0,1\}^*$ which is defined for every input size n as $f_n : x \mapsto C_n(x)$ is a quantum one-way function.*

Proof. For clarity, we are going to omit the parameter of the input size n from the inverter. Since, the circuit is efficient, one can implement the unitary map

$$U_f : |x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle \tag{5}$$

The theorem follows by proving the contrapositive. Assume that f is not a quantum one-way function. Then according to definition 5, for every polynomial p there exists a quantum circuit I' which succeeds in approximately inverting f, i.e. for all sufficiently large $n \in \mathbf{N}$ we have

$$I' : |f(x)\rangle|\beta\rangle \mapsto a_{f(x)}|f(x)\rangle|x \oplus \beta\rangle + b_{f(x)}|f(x)\rangle|G_{f(x)}\rangle \tag{6}$$

where $|G_{f(x)}\rangle$ is a garbage state, $\frac{1}{2^n}\sum_x a^2_{f(x)} > 1 - \frac{1}{p(n)}$ and the $a_{f(x)}$'s are positive. Now, from equations 5 and 6 we have:

$$\begin{array}{lll} |x\rangle|0\rangle & \to_{U_f} & |x\rangle|f(x)\rangle \\ & \to_{\text{SWAP}} & |f(x)\rangle|x\rangle \\ & \to_{I'} & a_{f(x)}|f(x)\rangle|0\rangle + b_{f(x)}|f(x)\rangle|G'_{f(x)}\rangle \end{array}$$

Starting with a uniform superposition of $x \in \{0,1\}^n$ we have

$$\frac{1}{2^{n/2}}\sum_{x\in\{0,1\}^n}|x\rangle|0\rangle \quad \to \quad \frac{1}{2^{n/2}}\sum_x(\ a_{f(x)}|f(x)\rangle|0\rangle + b_{f(x)}|f(x)\rangle|G'_{f(x)}\rangle\ ) \equiv |Q_n\rangle$$

We claim that the above circuit that on input $(|0\rangle, 1^n)$ outputs $|Q_n\rangle$ is a quantum sampler for C. Let $|C_n\rangle = \frac{1}{2^{n/2}}\sum_x |f(x)\rangle|0\rangle$ be the quantum sample of the circuit C, then

$$|\langle Q_n|C_n\rangle|^2 = |\frac{1}{2^n}\sum_x a_{f(x)}|^2 \geq |\frac{1}{2^n}\sum_x a^2_{f(x)}|^2 > (1 - 1/p(n))^2 > 1 - \epsilon.$$

This is a contradiction to C being a hard instance of the CQS problem and hence f is a quantum one-way function. □

4.2 Many-to-one one-way functions

The previous section dealt with the case of one-to-one one-way functions. Here, we generalize our results to the case of many-to-one functions. We show that the existence of a hard instance of CQS problem, where the circuit family $\{C_n\}$ is many-to-one, implies the existence of a quantum distributionally one-way function. In the next section we prove that a quantum distributionally one-way function implies a quantum one-way function.

Theorem 4 *Assume for a classical circuit family $\{C_n\}$, which computes a many-to-one function, the corresponding CQS problem is hard , i.e. there exists no efficient quantum circuit implementing QS_C. Then the function $f : \{0,1\}^* \to \{0,1\}^*$ which is defined for every input size n as $f_n : x \mapsto C_n(x)$ is a quantum distributionally one-way function.*

Proof. Since the classical circuit is efficient one can implement the unitary map

$$U_f : |x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$$

Assume that f is not a quantum distributional one-way, then according to definition 7 for every polynomial p there exists a quantum polynomial time algorithm S' which succeeds in approximately implementing a sampler for f, i.e. for all sufficiently large $n \in \mathbf{N}$ we have

$$S' : |f(x)\rangle|0\rangle \mapsto a_{f(x)}|f(x)\rangle|H_{f(x)}\rangle + b_{f(x)}|f(x)\rangle|G_{f(x)}\rangle \tag{7}$$

where $\frac{1}{2^n}\sum_{x\in\{0,1\}^n} a^2_{f(x)} > 1 - \frac{1}{p(n)}$ and the $a_{f(x)}$'s are positive.

Using the above unitaries, we can construct a quantum sampler QS_C that for every input n constructs a quantum sample for C_n:

$$\begin{aligned}
\sum_{x\in\{0,1\}^n} \frac{1}{2^{n/2}}|x\rangle|0\rangle \quad &\equiv \quad \sum_{f(x)} \frac{\sqrt{|f^{-1}(f(x))|}}{2^{n/2}}|H_{f(x)}\rangle|0\rangle \\
&\rightarrow_{U_f} \quad \sum_{f(x)} \frac{\sqrt{|f^{-1}(f(x))|}}{2^{n/2}}|H_{f(x)}\rangle|f(x)\rangle \\
&\rightarrow_{\text{SWAP}} \quad \sum_{f(x)} \frac{\sqrt{|f^{-1}(f(x))|}}{2^{n/2}}|f(x)\rangle|H_{f(x)}\rangle \\
&\rightarrow_{S'} \quad \sum_{f(x)} \frac{\sqrt{|f^{-1}(f(x))|}}{2^{n/2}}(\ a_{f(x)}|f(x)\rangle|0\rangle + b_f|f(x)\rangle|G_{f(x)}\rangle) \equiv |Q_n\rangle
\end{aligned}$$

The quantum sample for the circuit C_n is $|C_n\rangle = \sum_{f(x)} \frac{\sqrt{|f^{-1}(f(x))|}}{2^{n/2}}|f(x)\rangle|0\rangle$. Similarly to the proof of Theorem 3:

$$|\langle Q_n|C_n\rangle|^2 = |\sum_{f(x)} \frac{|f^{-1}(f(x))|}{2^n} a_{f(x)}|^2 = |\frac{1}{2^n}\sum_{x\in\{0,1\}^n} a_{f(x)}|^2 \geq |\frac{1}{2^n}\sum_{x\in\{0,1\}^n} a^2_{f(x)}|^2 > (1-1/p(n))^2 > 1-\epsilon.$$

This is a contradiction and hence, f is a quantum distributionally one-way function. □

4.3 From quantum distributionally one-way functions to quantum one-way functions

In the classical setting, Impagliazzo and Luby [9] proved that the existence of a distributionally one-way function implies the existence of a one-way function. In this section, we describe the main ideas of their construction and show how to prove the equivalent result in the quantum setting.

Theorem 5 *If there exists a quantum distributionally one-way function then there exists a quantum one-way function.*

4.3.1 The Impagliazzo-Luby construction

Let $f : \{0,1\}^* \to \{0,1\}^*$ be a candidate distributionally one-way function. Then, there exists a functions g such that an inverter I for g implies the existence of a sampler S for f. Without loss of generality the inverter for g outputs $\perp$ when it's given as input something which is not in the image of g.

Lets us fix the size of input to n, this can be done as we are working with a uniform circuit family. Now, in order to make the ideas of the construction clear, first assume that for a given $f(x)$ we know the size of the preimage $|f^{-1}(f(x))|$ and let $k = \lfloor \log |f^{-1}(f(x))| \rfloor + O(\log n)$. We define the function g as

$$g(x, h_k) = (f(x), h_k, h_k(x))$$

In other words, g takes as inputs an x and a random string h_k which can be thought of as a random hash function $h_k : \{0,1\}^n \to \{0,1\}^k$. The output of g is the value $f(x)$, the random hash function and the output of the hash function on x.

There are two observations to be made about the random hash function. First, since the range of the hash function is slightly larger than the number of x's in the preimage of $f(x)$, with high probability the mapping $x \mapsto h_k(x)$ for $\{x \in f^{-1}(f(x))\}$ is a one-to-one mapping. This implies, that if we could pick uniformly an element from the set $\{h_k(x) | x \in f^{-1}(f(x))\}$ then the inverter of g on input $(f(x), h_k, h_k(x))$ would return a uniform $x \in f^{-1}(f(x))$.

Second, it's indeed possible to pick a uniform element of the set $\{h_k(x) | x \in f^{-1}(f(x))\}$. Since the range of the hash function is not too much larger than the size of the preimage of $f(x)$, if we pick a random element $r_k \in \{0,1\}^k$, then with non negligible probability $r_k = h_k(x)$ for some $x \in f^{-1}(f(x))$.

The above two properties enable one to prove that, when one knows the size of the preimage of $f(x)$, the following procedure is a sampler for $f(x)$:

Partial Sampler PS(f(x),k)
Repeat a polynomial number of times
Pick a random hash function h_k and $r_k \in \{0,1\}^k$.
If $I(f(x), h_k, r_k) \neq \perp$ then output it and exit.
Output $\perp$

The only remaining issue is that the sampler doesn't know the size of the preimage of $f(x)$. Suppose we pick the range of the hash function to be much larger than the actual size of the preimage of $f(x)$. Then the above sampler outputs $\perp$ with very high probability. However, conditioned on it producing an output x, then this x is still almost uniformly distributed in $\{f^{-1}(f(x))\}$. This is true since the hash function randomly hashes $|f^{-1}(f(x))|$ values of x to a much larger range, and therefore, the mapping is with very high probability one-to-one.

Hence, we can construct a sampler for f by starting with the largest possible value for the range of the hash function and keep decreasing it until there is an outcome:

Sampler S(f(x))
For $j = n + O(\log n)$ to $O(\log)$:
If $PS(f(x), j) \neq \perp$ output it and exit.
Output $\perp$.

The analysis of the sampler for f is based on two observations. First, as we already said, if the

sampler produces an output for a $j \geq k$, then this x is guaranteed to be almost uniform. Second, the probability that the sampler actually produces an output for $j \geq k$ is, in fact, very close to 1. Impagliazzo and Luby make this argument rigorous in [9], taking also into account the fact that the inverter I of g is not perfect. Picking the right parameters in their construction, we have

Lemma 1 *[9] Let p_j be the probability that the Partial Sampler $PS(f(x), j)$ produces a legal output. Then, for all $j \geq k = \lfloor \log |f^{-1}(f(x))| \rfloor + \log n$*

$$(1 - o(1)) \left(1 - \left(\frac{1}{n}\right)^{2^{k-j}}\right) \leq p_j \leq 1 - \left(\frac{1}{n}\right)^{2^{k-j}}$$

4.3.2 The construction of the Quantum Sampler

Here, we reproduce the Impagliazzo-Luby construction in the quantum setting. As before, let $f : \{0,1\}^* \to \{0,1\}^*$ be the candidate quantum distributionally one-way function, fix the input size to be n, and define $g(x, h_k) = (f(x), h_k, h_k(x))$. Assuming that we have a quantum inverter I for g, our goal is to construct a quantum sampler for f, namely the following unitary

$$\text{QSampler: } |f(x)\rangle|0\rangle \mapsto a_{f(x)}|f(x)\rangle|H_{f(x)}\rangle + b_{f(x)}|G_{f(x)}\rangle,$$

where $\frac{1}{2^n}\sum_{x\in\{0,1\}^n} a^2_{f(x)} \geq 1 - o(1)$ and $|H_{f(x)}\rangle = \frac{1}{\sqrt{|f^{-1}(f(x))|}}\sum_{x\in f^{-1}(f(x))}|x\rangle$.

First, we assume that for a given $f(x)$ we know the size of the preimage $|f^{-1}(f(x))|$ and $k = \lfloor \log |f^{-1}(f(x))| \rfloor + O(\log n)$. The following unitary operations are the quantum equivalents of picking a random hash function h_k and a random string $r_k \in \{0,1\}^k$ and are efficiently constructible:

$$Q \;:\; |k\rangle|0\rangle \to |k\rangle \sum_{h_k} |h_k\rangle \quad , \quad B \;:\; |k\rangle|0\rangle \to |k\rangle \sum_{r_k} |r_k\rangle$$

Moreover, we describe the quantum inverter of g as

$$I \;:\; \left\{ \begin{array}{lcl} |f(x)\rangle|h_k\rangle|h_k(x)\rangle|0\rangle|0\rangle & \to_\epsilon & |f(x)\rangle|h_k\rangle|h_k(x)\rangle|x\rangle|0\rangle \\ |f(x)\rangle|h_k\rangle|s_k\rangle|0\rangle|0\rangle & \to & |f(x)\rangle|h_k\rangle|s_k\rangle|0\rangle|1\rangle \end{array} \right\}$$

where the random strings r_k have been divided into $h_k(x)$, which are the strings such that there is a unique $x \in f^{-1}(f(x))$ mapped to $h_k(x)$ and s_k, which are the strings for which there is no $x \in f^{-1}(f(x))$ mapped to s_k. The error parameter $\to_\epsilon$ accounts for the errors of the Inverter, including the cases where more than one x is mapped to the same $h_k(x)$. The last register in I acts as an "error flag". Last, recall that h_k is an efficient hash function and hence, having $|h_k\rangle$ and $|x\rangle$ one can efficiently compute $|h_k(x)\rangle$ and construct the following unitary:

$$T : |h_k\rangle|h_k(x)\rangle|x\rangle \quad \to \quad |h_k\rangle|0\rangle|x\rangle$$

We are now ready to define a partial quantum sampler for $f(x)$, when we know the size of its preimage. Denote by $p_{k,f(x)}$ the probability that the perfect inverter would return a legal output for given values of $f(x), k$. In the following, we drop the second subscript and have $p_k = p_{k,f(x)}$.

Partial Quantum Sampler PQS(f(x),k)

$$|f(x)\rangle|k\rangle|0\rangle|0\rangle|0\rangle|0\rangle$$

$$\overset{Q_3\otimes B_4}{\rightarrow} \quad |f(x)\rangle|k\rangle \sum_{h_k,r_k} |h_k\rangle|r_k\rangle|0\rangle|0\rangle \tag{i}$$

$$\overset{I_{1,3,4,5}}{\rightarrow_\epsilon} \quad \sqrt{p_k}|f(x)\rangle|k\rangle \sum_{h_k,h_k(x)} |h_k\rangle|h_k(x)\rangle|x\rangle|0\rangle + \sqrt{1-p_k}|f(x)\rangle|k\rangle \sum_{h_k,s_k} |h_k\rangle|s_k\rangle|0\rangle|1\rangle \tag{ii}$$

$$\overset{T_{3,4,5}}{\rightarrow} \quad \sqrt{p_k}|f(x)\rangle|k\rangle \sum_{h_k} |h_k\rangle|0\rangle \sum_{x\in f^{-1}(f(x))} |x\rangle|0\rangle + \sqrt{1-p_k}|f(x)\rangle|k\rangle \sum_{h_k,s_k} |h_k\rangle|s_k\rangle|0\rangle|1\rangle \tag{iii}$$

$$\overset{Q_3^\dagger}{\rightarrow} \quad \sqrt{p_k}|f(x)\rangle|k\rangle|0\rangle|0\rangle|H_{f(x)}\rangle|0\rangle + \sqrt{1-p_k}|f(x)\rangle|k\rangle|G_{f(x),k}\rangle|1\rangle \tag{iv}$$

where $|H_{f(x)}\rangle = \frac{1}{\sqrt{|f^{-1}(f(x))|}} \sum_{x\in f^{-1}(f(x))} |x\rangle$. In the first step, we construct a uniform superposition of all possible hash functions h_k and random strings $r_k \in \{0,1\}^k$. In the second step, we perform the Inverter of g. If the inverter was perfect and the mapping $x \mapsto h_k(x)$ was truly one-to-one, then the state would be exactly the one in (ii). The first term corresponds to the strings $r_k \in \{0,1\}^k$ such that $r_k = h_k(x)$ for a unique $x \in f^{-1}(f(x))$ and this happens with probability p_k. The second term corresponds to the rest of the strings. The error parameter ϵ accounts for the errors of the Inverter and the fact that the mapping is not exactly one-to-one. In the third step, we uncompute $h_k(x)$ and in the last step we uncompute the superposition of h_k. The final state in the perfect case consists of two terms. The first one is $|f(x)\rangle|k\rangle|H_{f(x)}\rangle$, where the third register contains a uniform superposition of the preimages of $f(x)$ and the second term denotes that the Sampler has failed ("error flag" register is 1). The norm of the first term is p_k, which is the probability that the inverter outputs a legal output for the given values $k, f(x)$.

Our partial quantum sampler imitates exactly the Impagliazzo-Luby one and hence we can use their analysis to show rigorously that conditioned on our sampler not failing, the actual state produced at the end is very close to the state $|f(x)\rangle|k\rangle|H_{f(x)}\rangle$. Moreover, since we picked $k = \lfloor \log |f^{-1}(f(x))| \rfloor + O(\log n)$ the norm of the term $|f(x)\rangle|k\rangle|H_{f(x)}\rangle$ is not negligible.

Though the classical and quantum partial samplers seem identical, there is, in fact, a difference. In the above procedure, for superposition inputs, different values of $|k\rangle$ and $|f(x)\rangle$ get entangled and so the naive way of implementing the classical sampler $S(f(x))$ as a quantum circuit will fail. This can be overcome by applying the classical procedure in a "clean" way *i.e.* garbage-free where the garbage in this case is the $|k\rangle$ register. However, since the classical procedure consists of a "While Loop" (a loop with an exit command) the procedure of un-computing the garbage is more demanding than the usual case where one deals with a "For Loop". To do so, instead of implementing the while loop of the classical algorithm we prepare a weighted superposition of all k's as an ancilla register which then leads to our garbage-free quantum sampler.

First we construct a partial ancilla preparation circuit for the case where the value of k is known. Basically, we apply our partial quantum sampler twice in order to "clean" the register that contains $|H(f(x))\rangle$, while copying the "error flag" in between.

Partial Ancilla Preparation, PAP(f(x),k)

$$
\begin{aligned}
& |f(x)\rangle|k\rangle|0\rangle|0\rangle|0\rangle \\
\overset{PQS(f(x),k)}{\rightarrow} \quad & \sqrt{p_k}|f(x)\rangle|k\rangle|H_{f(x)}\rangle|0\rangle|0\rangle + \sqrt{1-p_k}|f(x)\rangle|k\rangle|G_{f(x),k}\rangle|1\rangle|0\rangle \\
\overset{(\text{ctrl}-\text{NOT})_{4,5}}{\rightarrow_{\epsilon}} \quad & \sqrt{p_k}|f(x)\rangle|k\rangle|H_{f(x)}\rangle|0\rangle|0\rangle + \sqrt{1-p_k}|f(x)\rangle|k\rangle|G_{f(x),k}\rangle|1\rangle|1\rangle \\
\overset{PQS(f(x),k)^\dagger}{\rightarrow} \quad & \sqrt{p_k}\Big(\sqrt{p_k}|f(x)\rangle|k\rangle|0\rangle|0\rangle + \sqrt{1-p_k}|f(x)\rangle|k\rangle|00\rangle^\perp\Big)|0\rangle \\
+ \quad & \sqrt{1-p_k}\Big(\sqrt{1-p_k}|f(x)\rangle|k\rangle|0\rangle|0\rangle + \sqrt{p_k}|f(x)\rangle|k\rangle|00\rangle^\perp\Big)|1\rangle \\
= \quad & |f(x)\rangle|k\rangle|0\rangle|0\rangle\Big(p_k|0\rangle + (1-p_k)|1\rangle\Big) + |f(x)\rangle|k\rangle|00\rangle^\perp(\sqrt{p_k(1-p_k)}|0\rangle + \sqrt{p_k(1-p_k)}|1\rangle) \\
= \quad & |f(x)\rangle|k\rangle\Big(p_k|0\rangle + (1-p_k)|1\rangle\Big)|0\rangle + |f(x)\rangle|G_{f,k}\rangle|1\rangle
\end{aligned}
$$

In the last equation we have just rearranged the registers and put the "error flag" register at the end. We are now going to describe a circuit for the ancilla preparation when we start our algorithm for a large value of k and decrease it at each step by one. For clarity, the quantum registers contain the values n to 1 instead of $n + O(\log n)$ to $O(\log n)$ which are the real values for which the Sampler is run. Furthermore, all the operations are controlled by the "error flag" being the last register.

Ancilla Preparation AP(f(x))

$$
\begin{aligned}
& |f(x)\rangle|n\rangle|0\rangle|n-1\rangle|0\rangle\cdots|1\rangle|0\rangle|0\rangle \\
\overset{PAP_{1,2,3}}{\rightarrow} \quad & |f(x)\rangle|n\rangle\Big(p_n|0\rangle + (1-p_n)|1\rangle\Big)|n-1\rangle|0\rangle\cdots|1\rangle|0\rangle|0\rangle + |G\rangle|1\rangle \\
\overset{\text{ctr}_3-PAP_{1,4,5}}{\rightarrow} \quad & |f(x)\rangle|n\rangle p_n|0\rangle|n-1\rangle|0\rangle\cdots|1\rangle|0\rangle|0\rangle + \\
& |f(x)\rangle|n\rangle(1-p_n)|1\rangle|n-1\rangle\Big(p_{n-1}|0\rangle + (1-p_{n-1})|1\rangle\Big)\cdots|1\rangle|0\rangle|0\rangle + \\
& |G'\rangle|1\rangle \\
\overset{\text{ctr}_5-PAP_{1,6,7}}{\rightarrow} \quad & |f(x)\rangle|n\rangle p_n|0\rangle|n-1\rangle|0\rangle\cdots|1\rangle|0\rangle|0\rangle + \\
& |f(x)\rangle|n\rangle(1-p_n)|1\rangle|n-1\rangle p_{n-1}|0\rangle\cdots|1\rangle|0\rangle|0\rangle + \\
& |f(x)\rangle|n\rangle(1-p_n)|1\rangle|n-1\rangle(1-p_{n-1})|1\rangle|n-2\rangle\Big(p_{n-2}|0\rangle + (1-p_{n-2})|1\rangle\Big)\cdots|1\rangle|0\rangle|0\rangle + \\
& |G''\rangle|1\rangle \\
\vdots & \\
\rightarrow \quad & |f(x)\rangle|n\rangle\cdots|1\rangle\sum_j q_j|j\rangle|0\rangle + |G_f\rangle|1\rangle
\end{aligned}
$$

where $q_j = \prod_{i=1}^{j-1}(1-p_i)p_j$ is the probability that the sampler PQS succeeds at the j-th round and has failed on all previous rounds. Since the registers that contain the values n to 1 are not entangled with $f(x)$ we can ignore them and have

$$AP : |f(x)\rangle|0\rangle|0\rangle \mapsto |f(x)\rangle\sum_j q_j|j\rangle|0\rangle + |G_f\rangle|1\rangle$$

Now we can present the garbage-free quantum sampler for the general case where we don't know the size of the pre-image for a given $f(x)$. For clarity, we don't explicitly write down all the

necessary $|0\rangle$ registers in every step and also all the unitaries are performed when the "error flag" is 0.

Quantum Sampler, QS(f(x))

$$\begin{aligned}
& |f(x)\rangle|0\rangle|0\rangle \\
\stackrel{AP}{\rightarrow} \quad & |f(x)\rangle \sum_j q_j |j\rangle|0\rangle + |G^1_{f(x)}\rangle|1\rangle \\
\stackrel{PQS}{\rightarrow} \quad & |f(x)\rangle \sum_j q_j |j\rangle \Big(\sqrt{p_j}|H_{f(x)}\rangle|0\rangle + \sqrt{1-p_j}|G^2_{f(x),j}\rangle|1\rangle\Big)|0\rangle + |G^1_{f(x)}\rangle|1\rangle \\
= \quad & |f(x)\rangle \sum_j q_j \sqrt{p_j}|j\rangle|H_{f(x)}\rangle|0\rangle + |G^3_{f(x)}\rangle|1\rangle \\
\stackrel{AP^\dagger}{\rightarrow} \quad & \sum_j q_j^2 \sqrt{p_j}|f(x)\rangle|H_{f(x)}\rangle|0\rangle + |G^4_{f(x)}\rangle|1\rangle
\end{aligned}$$

where the last step follows from the unitarity of $AP^\dagger$, *i.e.* from

$$\begin{aligned}
|f(x)\rangle \sum_j q_j |j\rangle|0\rangle + |G^1_{f(x)}\rangle|1\rangle \quad & \stackrel{AP^\dagger}{\rightarrow} \quad |f(x)\rangle|0\rangle|0\rangle \\
|f(x)\rangle \sum_j q_j \sqrt{p_j}|j\rangle|0\rangle \quad & \stackrel{AP^\dagger}{\rightarrow} \quad \alpha|f(x)\rangle|0\rangle|0\rangle + \beta|G\rangle|1\rangle
\end{aligned}$$

we conclude that $\alpha = \Big(\langle f(x)| \sum_j q_j \langle j|\langle 0| + \langle G^1_{f(x)}|\langle 1|\Big)\Big(|f(x)\rangle \sum_j q_j \sqrt{p_j}|j\rangle|0\rangle\Big) = \sum_j q_j^2 \sqrt{p_j}$.

It remains to compute the success probability of the Garbage-free Quantum Sampler, i.e to calculate the square of the sum $\sum_j q_j^2 \sqrt{p_j}$. Proving that it is $1-o(1)$, then we obtain a contradiction to f being a quantum distributionally one-way function and hence we conclude that g is a quantum one-way function. Note that the success probability of the Impagliazzo-Luby sampler is $\sum_j q_j$ and Lemma 1 proves that for $j \geq k = \lfloor \log |f^{-1}(f(x))| \rfloor + O(\log n)$ one obtains $\sum_{j \geq k} q_j = 1 - o(1)$. Here, we have a slightly more complicated expression that can still be shown to be large.

Lemma 2 *The procedure QS is a quantum sampler for f with probability $1-o(1)$, i.e. $\sum_j q_j^2 \sqrt{p_j} \geq 1 - o(1)$.*

Proof. We are going to bound this sum by showing that there exists a particular m for which the term $q_m^2 \sqrt{p_m}$ is $1 - o(1)$. In order to do so, we slightly change the procedure we described above and instead of starting from $j = n + \log n$ and decreasing j at each step by 1, we pick a random offset $r \in [\log \log n]$, start with $j = n + \log n + r$ and decrease j at each step by $\log \log n$. Also, let $k = \lfloor \log |f^{-1}(f(x))| \rfloor + \log n$. The values of p_j for different j's can be estimated using Lemma 1

$$(1 - o(1)) \left(1 - \left(\frac{1}{n}\right)^{2^{k-j}}\right) \leq p_j \leq 1 - \left(\frac{1}{n}\right)^{2^{k-j}}$$

First, we bound the probability that the algorithm fails in all the rounds for $j = n + \log n + r$ to $j \geq k + (1 + \epsilon) \log \log n$, where for example $\epsilon = \frac{1}{\log \log \log n}$. Note that at each round j is decreased

by $\log\log n$. Since p_j is an decreasing function of j the minimum probability of failure is obtained for $r = 0$ and is

$$\begin{aligned}\prod_{j=k+(1+\epsilon)\log\log n}^{n+\log n}(1-p_j) &\geq \prod_{j=k+(1+\epsilon)\log\log n}^{n+\log n}\left(\frac{1}{n}\right)^{2^{k-j}} = \prod_{\ell\geq 1}\left(\frac{1}{n}\right)^{2^{-(\ell+\epsilon)\log\log n}} \\ &= \prod_{\ell\geq 1}\left(\frac{1}{n}\right)^{(\log n)^{-(\ell+\epsilon)}} = \left(\frac{1}{n}\right)^{\sum_{\ell\geq 1}(\log n)^{-(\ell+\epsilon)}} \\ &\approx \left(\frac{1}{n}\right)^{\frac{1}{(\log n)^{1+\epsilon}-1}} = 1-o(1)\end{aligned}$$

Moreover, for any $j \in [k+\epsilon\log\log n, k+(1-\epsilon)\log\log n,]$ we have that

$$p_j \geq 1-\left(\frac{1}{n}\right)^{2^{-(1-\epsilon)\log\log n}} = 1-\left(\frac{1}{n}\right)^{(\log n)^{-(1-\epsilon)}} = 1-\left(\frac{1}{2}\right)^{(\log n)^{\epsilon}} = 1-o(1)$$

Since we pick a random initial offset $r \in [\log\log n]$, then with probability $(1-2\epsilon)$ over r the algorithm is run for an $m \in [k+\epsilon\log\log n, k+(1-\epsilon)\log\log n]$. In this case, we have already shown that $p_m = 1-o(1)$ and, moreover, for all previous rounds we have $j \geq k+(1+\epsilon)\log\log n$ and hence the probability of failure is $\prod_{j>m}(1-p_j) = 1-o(1)$. To sum up, with probability $(1-2\epsilon) = 1-o(1)$ our algorithm is run for an m such that

$$\sum_j q_j^2\sqrt{p_j} \geq q_m^2\sqrt{p_m} = \prod_{j>m}(1-p_j)^2 p_m^{5/2} = 1-o(1)$$

and therefore the overall success probability of the algorithm is $1-o(1)$. □

This concludes the proof of Theorem 5 and together with Theorem 4 we have

Theorem 6 *Assume for a classical circuit C, which computes a many-to-one function, the corresponding CQS problem is hard , i.e. there exists no poly($|C|$) size quantum circuit implementing QS_C. Then there exists a quantum one-way function.*

5 Statistical Zero Knowledge and quantum one-way functions

The CQS problem has an interesting connection to the classical complexity class of Statistical Zero Knowledge (SZK) languages:

Theorem 7 *[2] Any language $\mathcal{L} \in$ SZK can be reduced to a set of instances of the CQS problem.*

The proof is based on a reduction of the following SZK-complete problem to a quantum sampling problem.

Definition 8 *[16] Consider two constants $0 \leq \beta < \alpha \leq 1$ such that $\alpha^2 > \beta$. Statistical Difference ($SD_{\alpha,\beta}$) is the promise problem of deciding for any two given classical circuits C_0 and C_1 whether their output distributions are close to or far from each other, i.e. whether:*

$$||D_{C_0} - D_{C_1}|| \geq \alpha \quad or \quad ||D_{C_0} - D_{C_1}|| \leq \beta$$

It's not hard to see that the above problem can be reduced to the problem of quantum sampling the circuits C_0 and C_1. Indeed, if one could efficiently construct the quantum samples $|C_0\rangle$ and $|C_1\rangle$, then, by performing a SWAP-test, one could decide whether the two circuit distributions are close to or far from each other. Equivalently, the above problem can be reduced to the problem of quantum sampling the circuit $C \triangleq C_0 \otimes C_1$, since a SWAP-test would again decide whether the two circuit distributions are close or far. Based on this result, we obtain the quantum analog of Ostrovsky's result [15]:

Theorem 8 *Assume there exists a language $\mathcal{L} \in \mathrm{SZK} \setminus \mathrm{AvgBQP}$, then quantum one-way functions exist.*

Proof. Assume $\mathcal{L} \in \mathrm{SZK} \setminus \mathrm{AvgBQP}$. For every input size n, let $\{C^x\}_{x\in\{0,1\}^n}$ be the set of classical circuits which decide L via reduction to the complete language in Definition 8. Denote by $m = poly(n)$ the size of the input to the circuits from this set. Since the language $\mathcal{L}$ is not in AvgBQP, for any sufficiently large input size n, there exists a samplable distribution $\mathcal{D}_n$ such that for $x \sim \mathcal{D}_n$, the language $\mathcal{L}$ can not be decided with high probability with a polynomial time quantum algorithm. Equivalently there is no polynomial quantum algorithm that produces a quantum sample of C^x for an average $x \sim \mathcal{D}_n$. We can assume this distribution to be uniform [8] and hence we have a uniform family of sets of circuits $\{\{C^x\}_{x\in\{0,1\}^n}\}_{n\in\mathbf{N}}$, such that for any polynomial time quantum algorithm Q, any constant $\epsilon \in [0, 1/2)$, and all sufficiently large $n \in \mathbf{N}$

$$Q : |x\rangle|0\rangle \mapsto c_x|x\rangle|C^x\rangle + d_x|G_x\rangle$$

with

$$\frac{1}{2^n}\sum_x |(Q(|x\rangle|0\rangle)\ ,\ |x\rangle|C^x\rangle)|^2 = \frac{1}{2^n}\sum_x |c_x|^2 < 1-\epsilon$$

We define the function $f_C : \{0,1\}^* \to \{0,1\}^*$ such that $f_C : (x, y) \mapsto (x, C^x(y))$ and prove that it is a quantum one-way function. We assume that f is one-to-one otherwise from Theorem 5, we can obtain the same result. Suppose that the function f_C is not one-way, then there exists an inverter such that

$$I : |f(x,y)\rangle|0\rangle|0\rangle \mapsto a_{f(x,y)}|f(x,y)\rangle|x\rangle|y\rangle + b_{f(x,y)}|G_{f(x,y)}\rangle,$$

or equivalently

$$I : |x\rangle|C^x(y)\rangle|0\rangle \mapsto a_{f(x,y)}|x\rangle|C^x(y)\rangle|y\rangle + b_{f(x,y)}|G_{f(x,y)}\rangle,$$

where $\frac{1}{2^{n+m}}\sum_{x,y} a^2_{f(x,y)} \geq 1 - \frac{1}{p(n)}$ (the average is taken over x, y) and the $a_{f(x,y)}$'s are positive. We start from a uniform superposition of all y and use the inverter to create a circuit that is a good-on-average quantum sampler (similar to the proof of Theorem 3):

$$\begin{array}{lll} |x\rangle\frac{1}{2^{m/2}}\sum_y |y\rangle|0\rangle & \to_{U_f} & |x\rangle\frac{1}{2^{m/2}}\sum_y |y\rangle|C^x(y)\rangle \\ & \to_{\mathrm{SWAP}} & |x\rangle\frac{1}{2^{m/2}}\sum_y |C^x(y)\rangle|y\rangle \\ & \to_I & |x\rangle\frac{1}{2^{m/2}}\sum_y (a_{f(x,y)}|C^x(y)\rangle|0\rangle + b_{f(x,y)}|G'_{f(x,y)}\rangle) \equiv |x\rangle|T_m\rangle \end{array}$$

and hence for an average x

$$\begin{aligned} \frac{1}{2^n}\sum_x |\langle x|\langle T_m||x\rangle|C^x\rangle|^2 &= \frac{1}{2^n}\sum_x |\frac{1}{2^m}\sum_y a_{f(x,y)}|^2 \geq |\frac{1}{2^{n+m}}\sum_{x,y} a_{f(x,y)}|^2 \\ &\geq |\frac{1}{2^{n+m}}\sum_{x,y} a^2_{f(x,y)}|^2 \geq (1-\frac{1}{p(n)})^2 \geq 1-\epsilon \end{aligned}$$

This is a contradiction and hence the function f_C is a quantum one-way. □

6 Conclusions

In this paper we prove that the existence of any problem in SZK which is hard-on-average for a quantum computer, implies the existence of quantum one-way functions. Our proofs go through the problem of quantum sampling. Aharonov and Ta-Shma cast many important problems as quantum sampling problems and described a possible way for attacking them. It is, hence, very interesting to investigate the real hardness of quantum sampling. We already know that if SZK $\not\subseteq$ AvgBQP then there exist hard instances of quantum sampling. Under what other assumptions can one prove the existence of hard instances of the CQS problem and consequently quantum one-way functions?

Furthermore, we saw that our candidate one-way problems include some of the most notorious problems in quantum computing, like Graph Non-Isomorphism and approximate Closest Lattice Vector problem. Could we construct one-way functions from other problems, such as the hidden subgroup problem in the dihedral or other non-abelian groups?

Last, Watrous [18] proved that computational zero knowledge for NP is implied by the existence of quantum one-way permutations. What other implications does the existence of quantum one-way functions have?

Acknowledgements

We wish to thank Oded Regev, Ben Reichardt and Alain Tapp for useful discussions and Andrej Bogdanov for sharing with us his notes on [9]. EK was partially supported by the ARDA, MITACS, ORDCF, and CFI projects. IK gratefully acknowledges professor Peter w. Shor who has supported his research from a scholarly allowance provided from his appointment as the holder of the Henry Adams Morss and Henry Adams Morss, Jr. Professorship and from NSF CCF-0431787.

References

[1] M. Adcock and R. Cleve. A quantum goldreich-levin theorem with cryptographic applications. In *Proceedings of 19th Annual Symposium on Theoretical Aspect of Computer Sciences*, page 323, 2002.

[2] D. Aharonov and A. Te-shma. Adiabatic quantum state generation and statistical zero knowledge. In *Proceedings of STOC'02 – Symposium on the Theory of Computing*. ACM, 2001.

[3] C. Crépeau, F. Légaré, and L. Salvail. How to convert the flavor of a quantum bit commitment. In *Advances in Cryptology — EUROCRYPT 2001*, 2001.

[4] P. Dumais, D. Mayers, and L. Salvail. Perfectly concealing quantum bit commitment from any one-way permutation. In B. Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, number 1807 in Lecture Notes in Computer Science, Berlin — Heidelberg — New York, 2000. Springer Verlag.

[5] O. Goldreich. *Foundations of Cryptography - Volume 1*. Cambridge University Press, 2001.

[6] D. Gottesman and I. Chuang. Quantum digital signatures. quant-ph/0105032, 2001.

[7] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM journal of Computing*, 17:309, 1988.

[8] R. Impagliazzo and L. A. Levin. No better ways to generate hard np instances than picking uniformly at random. In *Proceedings of FOCS'90 – Symposium on Foundations of Computer Science*, 1990.

[9] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *Proceedings of FOCS'89 – Symposium on Foundations of Computer Science*, 1989.

[10] E. Kashefi, H. Nishimura, and V. Vedral. On quantum one-way permutations. *Quantum Information and Computation*, 2:379, 2002.

[11] A. Kawachi, H. Kobayashi, T. Koshiba, and R. H. Putra. Universal test for quantum one-way permutations. In *Proceedings of MFCS'04 – The 29th International Symposium on Mathematical Foundations of Computer Science*, 2004.

[12] A. Kitaev and J. Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof systems. In *Proceedings of STOC'00 – Symposium on the Theory of Computing*, page 608, 2000.

[13] E. Knill. Quantum randomness and nondeterminism. *arXiv.org e-Print quant-ph/9610012*, 1996.

[14] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2000.

[15] R. Ostrovsky One-way functions, hard on average problems and Statistical Zero Knowledge proofs. *IEEE Conference on Structure in Complexity Theory*, 1991

[16] A. Sahai and S. P. Vadhan. A complete promise problem for Statistical Zero Knowledge. In *Proceedings of FOCS'97 – Symposium on Foundations of Computer Science*, page 448, 1997.

[17] J. Watrous. Succinct quantum proofs for properties of finite groups. In *Proceedings of FOCS'2000 – Symposium on Foundations of Computer Science*, page 537, 2000.

[18] J. Watrous. Zero knowledge against quantum attacks. quant-ph/0511020.

Post-quantum multivariate-quadratic public key schemes

Jacques Stern

École Nationale Supérieure

Since the invention of public key cryptography by Diffie and Hellman in 1976, very few public key schemes have been deployed in applications, besides the celebrated RSA algorithm designed by Rivest, Shamir, and Adleman. While millions of RSA keys are used in WEB browsers, most public key cryptosystems are only present in textbooks, despite the fact that, if ever it happens, the advent of quantum computers would render RSA insecure.

There are however, several challenging lines of research proposing public key schemes of a different flavour. Among these is multivariate cryptography, which stems from considering the mathematical formula describing RSA as a univariate modular polynomial, and attempts to use multivariate polynomials instead, hoping that the cost of encryption and/or decryption will be lower.

While the original proposals based on this idea have been shown insecure, many further schemes have been designed. In turn, these have been subject to active cryptanalytic work. The aim of the talk is to review some of the schemes, to explain some of the methods that have been used to attack them, and to assess their level of security. In other words, is multivariate cryptography now ready for practical applications? Or do we have to wait until quantum computers are built?

Probabilistic Multivariate Cryptography

Aline Gouget*[1] and Jacques Patarin[2]

[1] France Telecom R&D, 42 rue des Coutures, F-14000 Caen, France.
[2] University of Versailles, 45 avenue des Etats-Unis, F-78035 Versailles, France.

Abstract. In public key schemes based on multivariate cryptography, the public key is a finite set of m (generally quadratic) polynomial equations and the private key is a trapdoor allowing the owner of the private key to invert the public key. In existing schemes, a signature or an answer to an authentication is valid if *all* the m equations of the public key are satisfied. In this paper, we study the idea of *probabilistic multivariate cryptography*, i.e., a signature or an authentication value is valid when at least α equations of the m equations of the public key are satisfied, where α is a fixed parameter of the scheme (or more generally when at least α_1 of the first m_1 equations of the public key are satisfied, and at least α_2 of the m_2 next equations of the public key are satisfied, etc., and at least α_ℓ of the last m_ℓ equations are satisfied, where $\alpha_1, \ldots, \alpha_\ell, m_1, \ldots, m_\ell$ and ℓ are well chosen integers with $m_1 + \cdots + m_\ell = m$). We show that many new public key signature and authentication schemes can be built using this concept. We apply this concept on some known multivariate schemes and we show how it can improve the security of the schemes.

1 Introduction

The security of most of the public key schemes relies on the difficulty of solving one of the two problems that are currently considered to be hard, i.e., the problem of factoring large integers and the problem of computing discrete logarithms. However, the techniques for solving these two famous problems improve continually. Then, it becomes very important to find alternative problems and to proceed further to the study of known candidates that are considered to be minors until now. Furthermore, some new attractive properties may be achieved by using alternative difficult problems.

One possibility for secure public key schemes is based on the problem of solving *multivariate nonlinear equations* over small finite fields. In multivariate cryptography, the public key is a set $\mathcal{A}$ of m polynomial equations in n variables over a small finite field K. Public key schemes for encryption, signature or authentication can be built with such public keys. Most of the time, the equations are chosen quadratic since solving quadratic systems is already $\mathcal{NP}$-complete and also hard on average.

* New affiliation: Gemplus, 34 rue Guynemer, F-92447 Issy-les-Moulineaux, France.

1.1 Related work

Since the introduction of the first multivariate schemes [5, 13, 7] in 1985, many schemes have been proposed. Most of these schemes have been broken but several schemes are still unbroken. Recently, C. Wolf and B. Preneel proposed a taxonomy [23] of public key schemes based on the problem of multivariate quadratic equations. They grouped the known schemes into a taxonomy of only four schemes: Matsumoto Imai (C^*) [13], Hidden Field Equations (HFE) [16], Stepwise Triangular Systems (STS) [22] and Unbalanced Oil and Vinegar (UOV) [8].

Some of these schemes [13, 22] are broken. However, from these four basic schemes, it is possible to design more schemes by applying a *perturbation* in order to improve the security of the basic scheme. For instance, the scheme C^{*--} which is a variant of the C* scheme using the perturbation *minus* (i.e., a part of the public key is kept secret) is still unbroken. Recently, J. Ding [4] proposed another variant of the C^* cryptosystem using a new perturbation called *internal perturbation* which was next broken by P-A. Fouque et al. [6]. However the internal perturbation can be applied on HFE for instance.

The security of the unbroken schemes is most of the time an open problem since it consists in checking that all known attacks do not apply. However, multivariate schemes have attractive properties that cannot be reached using classical public key schemes based on factorization or discrete logarithm. For instance, it becomes possible to get very short signatures or very fast computations. Furthermore, the study of multivariate schemes is interesting from a theoretical point of view since it leads to the study of some new specific problems.

A notion close to the idea of *probabilistic multivariate cryptography* presented in this paper is given in [1] but the context is different since it is the traitor tracing based on the IP problem.

1.2 Outline

In Section 2, we first present the general problem of multivariate polynomials and the public key of multivariate schemes. Then, we compare how this public key is used in classical (non-probabilistic) schemes and in probabilistic schemes. In Section 3, we explain how a probabilistic scheme can be built from a classical trapdoor. This construction will sometimes also hide the trapdoor in a much better way than in a classical construction. In Sections 4 and 5, we present some explicit probabilistic multivariate schemes: in Section 4, we present an adaptation of the multivariate scheme C^* in a probabilistic way and we discuss several variants of the proposed scheme, and in Section 5, we present an adaptation of the multivariate scheme UOV. Finally, we conclude in Section 6.

2 Public key of multivariate schemes

In this section, we first recall the general difficult problem underlying multivariate cryptography. Next, we briefly describe public key schemes (i.e. authentication, signature and public key encryption schemes) in the context of *classical*

multivariate cryptography (i.e. the multivariate cryptography of the state of the art). Then, we describe the public key protocols in the context of *probabilistic* multivariate cryptography.

2.1 Problem of polynomial equations in finite fields

Let K be a finite field. Let $\mathcal{A} = (a_1, \ldots, a_m)$ be a system of $m \in \mathbb{N}$ polynomial equations in $n \in \mathbb{N}$ variables with degree $d \in \mathbb{N}$. Given $y = (y_1, \ldots, y_m) \in K^m$, the problem is to find a solution $x = (x_1, \ldots, x_n) \in K^n$ of the system:

$$\begin{cases} y_1 & = a_1(x_1, \ldots, x_n) \\ y_2 & = a_2(x_1, \ldots, x_n) \\ & \vdots \\ y_m & = a_m(x_1, \ldots, x_n) \end{cases}$$

Most of the time, the polynomial equations of a multivariate cryptographic scheme are quadratic (i.e. $d = 2$) since the problem of solving such system is $\mathcal{NP}$-complete and hard on average. In this case, the problem is called *Multivariate Quadratic Equations* problem and for every i, $1 \leq i \leq m$, the polynomial a_i has the form:

$$a_i = \sum_{1 \leq j \leq n} \sum_{1 \leq k \leq n} \gamma_{i,j,k} x_j x_k + \sum_{1 \leq j \leq n} \delta_{i,j} x_j + \xi_i \ ,$$

where the coefficients $\gamma_{i,j,k}$, $\delta_{i,j}$ and ξ_i are elements of K.

2.2 Classical multivariate schemes

A classical multivariate scheme relies on the knowledge of a trapdoor $T_{\mathcal{A}}$ in connection with a system $\mathcal{A}$ of m polynomial equations in n variables over a finite field K. Then:

- the public key is the system $\mathcal{A}$;
- the private key is the trapdoor $T_{\mathcal{A}}$ that allows to compute, for any given value $y = (y_1, \ldots, y_m)$, a value $x = (x_1, \ldots, x_n)$ such that, $y_i = a_i(x_1, \ldots, x_n)$ for every i, $1 \leq i \leq m$ (or equivalently such that $y = \mathcal{A}(x)$).

On the one hand, the computation of x such that $y = \mathcal{A}(x)$ must be easy using the trapdoor $T_{\mathcal{A}}$, and on the other hand, the computation of x without the knowledge of the trapdoor $T_{\mathcal{A}}$ must be computationally difficult (i.e. the number of operations must be greater than 2^{80}).

Multivariate signature Given a message M, one can compute the hash value y of the message M, i.e. $y = H(M)$, where H is a collision resistant hash function. Then, given a hash value y of a message M, a *signature* of the message M is a value x such that $y = \mathcal{A}(x)$. Only the owner of the private key can compute such a value x, and any verifier can check that $y = \mathcal{A}(x)$ for the hash value y of a given message M, its signature x and the public key $\mathcal{A}$.

Multivariate authentication An *authentication* between a prover and a verifier works as follows. The verifier sends a challenge y to the prover. Then, by using the trapdoor $T_{\mathcal{A}}$, the prover computes the value x such that $y = \mathcal{A}(x)$, and he sends x to the verifier. At last, the verifier computes $\mathcal{A}(x)$ and the authentication protocol is valid if and only if the equality $y = \mathcal{A}(x)$ holds.

Multivariate public key encryption For an *encryption scheme*, anybody can encrypt a message x by using the public key $\mathcal{A}$, that is, anybody can computes the ciphertext $y = \mathcal{A}(x)$. Furthermore, only the owner of the private key $T_{\mathcal{A}}$ can decrypt the value $y = \mathcal{A}(x)$ and recovers the value x.

Then, in classical multivariate schemes, *all* the m equations of the system $y = \mathcal{A}(x)$ must be satisfied in order to validate a protocol.

2.3 Probabilistic multivariate schemes

In this paper, we focus on authentication protocols and signature schemes (it may also be possible to build probabilistic encryption scheme but this is a difficult problem that we will not study here).

In a *probabilistic multivariate scheme*, the public key is a system $\mathcal{A}$ of m polynomial equations in n variables. A signature (resp. a response to a challenge) will be valid if **at least** α equations of the system $\mathcal{A}$ are satisfied where α is a fixed parameter of the scheme (or more generally, if at least α_1 of the m_1 first equations of $\mathcal{A}$ are satisfied, and at least α_2 equations of the m_2 next equations of $\mathcal{A}$ are satisfied etc., and at least α_ℓ of the m_ℓ last equations of $\mathcal{A}$ are satisfied, where $\alpha_1, \ldots, \alpha_\ell, m_1, \ldots, m_\ell$ and ℓ are well chosen integers with $m_1 + \cdots + m_\ell = m$).

General description when $m_1 = m$. Let K be a finite field (generally, we have $K = GF(2)$). The public key $\mathcal{A}$ is a system of m polynomial equations in n variables,

$$\mathcal{A} = \begin{cases} y_1 = a_1(x_1, \ldots, x_n) \\ y_2 = a_2(x_1, \ldots, x_n) \\ \vdots \\ y_m = a_m(x_1, \ldots, x_n) \end{cases}$$

where $x_1, \ldots, x_n, y_1, \ldots, y_m$ are variables defined over K and $a_1, \ldots, a_m$ are polynomials of degree d with coefficients in K.

The construction of a probabilistic multivariate scheme relies on the existence of a trapdoor $T_{\mathcal{A}}$ such that, given a value y, it is possible with a probability close to 1, to find a value x such that at least α equations of the m equations of $\mathcal{A}$ are satisfied. The parameter α is fixed; for instance, if $K = GF(2)$, then we have $\alpha > \frac{m}{2}$. In exchange, the probability to find a value x (such that α equations of $\mathcal{A}$ are satisfied) without the knowledge of $T_{\mathcal{A}}$ must be very close to 0.

Assuming that such a trapdoor exists, one can construct a probabilistic multivariate scheme for signature or authentication:

- In an authentication protocol, the verifier generates a random value y called a challenge and sends it to the prover.
- In a signature scheme, y is the hash value of the message M to be signed (i.e. $y = H(M)$) and x is the signature of y. In the following, we assume that the hash function H is not only collision resistant but also near-collision resistant, i.e., we assume that it is difficult to find y and y' such that $H(y) \oplus H(y')$ has low Hamming weight[3].

A value x such that at least α equations of the m equations of $\mathcal{A}$ are satisfied is a valid authentication value or a valid signature. Then, given a challenge (resp. a hash value of a message) y, the verification of the authentication (resp. the signature) x consists in verifying that the number of equations of $\mathcal{A}$ for the pair value (x, y) which are satisfied is greater than or equal to α.

The motivation for studying probabilistic multivariate cryptography are two-fold. First, many multivariate schemes have been broken. Sometimes, modifiers or perturbations are applied on these basic schemes in order to get secure multivariate schemes. Probabilistic multivariate cryptography is an alternative way to get secure multivariate schemes from basic known trapdoors. Second, there are few known basic trapdoors in the context of classical multivariate cryptography. Furthermore, the existence of many more basic trapdoors is uncertain. It would be very interesting (but certainly very difficult) to find new basic trapdoors and it may be easier to find a basic trapdoor for probabilistic multivariate schemes than for classical multivariate schemes.

In this paper, we only consider the construction of probabilistic multivariate schemes based on known trapdoors.

3 Design of a probabilistic multivariate schemes from a classical trapdoor

Let $\mathcal{A}$ denote the public key of the probabilistic multivariate scheme that we will build ($\mathcal{A}$ will be a system of m polynomials equations in n variables). Let $\mathcal{B}$ denote the public key of a classical multivariate scheme (i.e. $\mathcal{B}$ is a system of m polynomials equations in n variables) and let $T_{\mathcal{B}}$ denote the trapdoor associated to $\mathcal{B}$.

3.1 General construction

Construction of the public key $\mathcal{A}$. For simplicity, we set the finite field K to be $GF(2)$. Recall that $\mathcal{B} = (b_1, \ldots, b_m)$ is a system of $m \in \mathbb{N}$ polynomial equations in $n \in \mathbb{N}$ variables with degree $d \in \mathbb{N}$ and $T_{\mathcal{B}}$ denotes the trapdoor associated to $\mathcal{B}$. Let $\mathcal{C} := (c_1, \ldots, c_m)$ be a system of $m \in \mathbb{N}$ polynomial equations

[3] Assuming this additional condition on the hash function H is one technique to avoid existential forgery; alternative techniques will be presented in the full version of this paper.

in $n \in \mathbb{N}$ variables such that $c_i(x_1, \ldots, x_n) = 0$ with probability κ, where $\kappa > \frac{1}{2}$. Then, as we will see, we will obtain a probabilistic scheme with a public key $\mathcal{A}$ of the form:

$$\mathcal{A} = \begin{cases} y_1 = b_1(x_1, \ldots, x_n) & + & c_1(x_1 \ldots, x_n) & = a_1(x_1, \ldots, x_n) \\ y_2 = b_2(x_1, \ldots, x_n) & + & c_2(x_1 \ldots, x_n) & = a_2(x_1, \ldots, x_n) \\ & \vdots & & \\ y_m = b_m(x_1, \ldots, x_n) & + & c_m(x_1 \ldots, x_n) & = a_m(x_1, \ldots, x_m) \end{cases}$$

Remark 1. The system $\mathcal{B}$ can be constructed using any basic trapdoor and the polynomials c_i can be seen as perturbations of this basic trapdoor. In other words, the system $\mathcal{C}$ allows to *mask* the algebraic structure of the system $\mathcal{B}$. In Section 4, we use the C^* scheme and in Section 5, we use the Oil and Vinegar system. It is also possible to use for example a FLASH scheme, i.e. the C^{*--} scheme [18] or the HFE scheme [16].

We now describe the general execution of a probabilistic multivariate authentication protocol based on a known trapdoor. We do not describe the general execution of a probabilistic multivariate signature scheme based on a known trapdoor. We only mention that one solution is to assume the knowledge of a near-collision hash function H and to replace the challenge y sent by the verifier into the authentication protocol by the hash value $y = H(M)$ of the message M to be signed; alternative solutions will be presented in the full version of this paper.

Authentication scheme

1. The verifier randomly chooses a challenge $y = (y_1, \ldots, y_m)$ in $\in K^m$ and sends it to the prover.
2. The prover follows three steps:
   (a) For every $i \in [1; m]$, the value y_i is replaced by $y_i \oplus 1$ with probability β (where β is a fixed parameter such that $\beta \neq 0$ [4]), i.e. we have $y'_i = y_i$ with probability $1 - \beta$ and $y'_i \neq y_i$ with probability β. Then, the prover gets the value $y' = (y'_1, \ldots, y'_m)$. In average, βm values of y are modified to get y'.
   (b) Using the trapdoor $T_{\mathcal{B}}$, the prover computes the value $x = (x_1, \ldots, x_n)$ such that for every $i \in [1; m]$, we have
   $$y'_i = b_i(x_1, \ldots, x_n)$$
   (c) The prover checks that for at least α integers i of $[1; m]$, the equation
   $$y_i = b_i(x_1, \ldots, x_n) + c_i(x_1, \ldots, x_n)$$
   is satisfied. If not, then the prover restart at the beginning of step 2, else the prover sends $x = (x_1, \ldots, x_n)$ to the verifier.

[4] the reason why β must be different from 0 will be explained in Section 3.2

3. Finally, the verifier checks that at least α equations of the system :

$$\begin{cases} y_1 \stackrel{?}{=} a_1(x_1, \ldots, x_n) \\ \vdots \\ y_m \stackrel{?}{=} a_m(x_1, \ldots, x_n) \end{cases}$$

are satisfied.

The general execution of a probabilistic scheme is summarized in Figure 1.

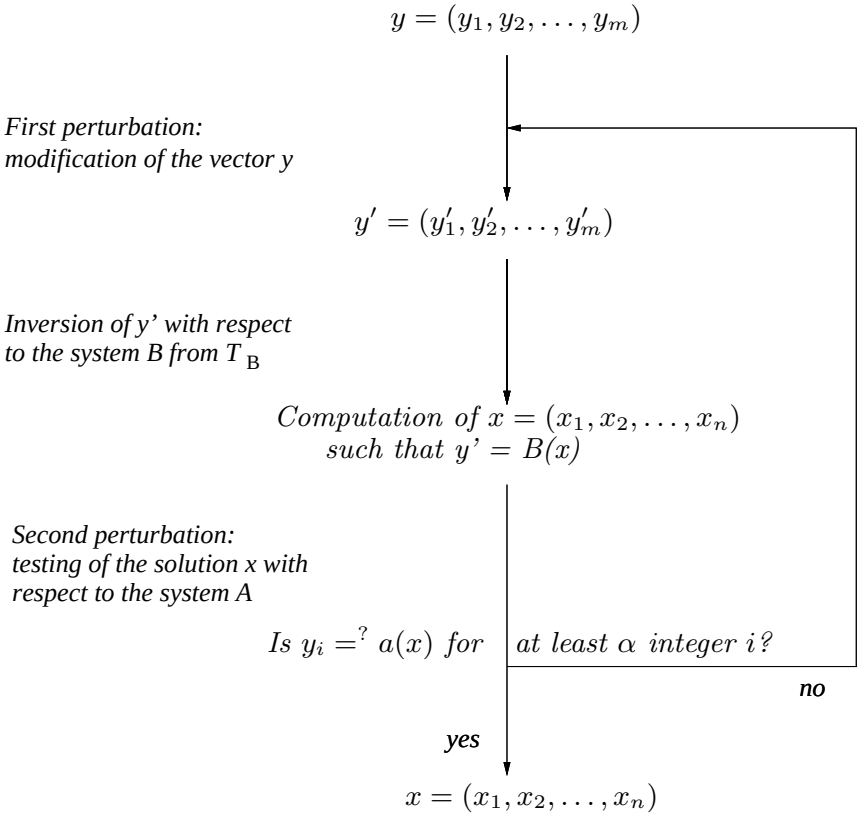


Fig. 1. Example of a probabilistic scheme

Remark 2. In practice, the indices i such that $y_i \neq y'_i$ are chosen with a pseudo-random algorithm that depends only of $(y_1, \ldots, y_m)$ such that for every i, $1 \leq i \leq m$, we have $y_i \neq y'_i$ with probability β and of the current run. Then, if the challenge $y = (y_1, \ldots, y_m)$ is given twice, then the prover will always answer with the same $x = (x_1, \ldots, x_n)$. Here the aim is to prevent the attacker from replaying the same challenge several times in order to get information on the system $\mathcal{C}$.

Remark 3. The value of the parameter κ is fixed by choosing the polynomials c_i, $1 \leq i \leq m$ (for instance, in Section 4, the quadratic polynomials c_i are chosen such that $\kappa = \frac{3}{4}$); the values of the parameters α and κ are discussed in the following of the section.

Remark 4. In the signature scheme, y is the hash value of the message M, i.e. $y = H(m)$, where H is near-collision resistant. This condition on H is to avoid the following attack. Assume that $(M, y = H(M), x)$ is a valid tuple such that there are $\alpha+a$ equations satisfied with $a > 0$. Then, one can construct a new pair (y', x) by changing up to a component in y. Thus, if H is not near-collision resistant, then an attacker will be able to construct a valid tuple $(M', y' = H(M'), x)$.

3.2 The parameter β must be different from 0

Recall that β is the probability that a bit y_i of the received challenge y is modified by the prover (before inverting the system). The role of the perturbation system $\mathcal{C}$ is to mask the algebraic structure of the system $\mathcal{B}$ (the aim is to prevent the attacker from accessing the system $\mathcal{B}$). However, in order to prevent the attacker to reconstruct the system $\mathcal{C}$, and then, to retrieve the system $\mathcal{B}$, the parameter β must be chosen in a better way.

Suppose that $\beta = 0$. Then, for every pair (x, y) the attacker would know that all the equations of $\mathcal{B}$ are satisfied by (x, y) with probability 1. If $\beta = 0$, then from $\mathcal{O}(n^2)$ pairs (x, y), an attacker will be able to reconstruct the system $\mathcal{B}$ with probability 1 by Gaussian reductions (on the quadratic coefficients of the equations of $\mathcal{B}$). In this case the difficulty of breaking the system is equivalent to the difficulty of breaking the original trapdoor associated to the system $\mathcal{B}$. Thus $\mathcal{C}$ has no interest anymore since it can be removed. So β must be different from zero.

When β is different from zero, the attacker has to deal with several cases:

- if a relation $y_i = a(x)$ is valid, then:
  1. y_i equals y'_i and $c_i(x) = 0$ happens with probability $(1 - \beta)(1 - \kappa)$ (on average);
  2. y_i is different from y'_i and $c_i(x) = 1$ happens with probability $\beta\kappa$ (on average);
- if a relation y_i is different from $a(x)$, then:
  1. y_i equals y'_i and $c_i(x) = 1$ happens with probability $(1-\beta)\kappa$ (on average);
  2. y_i is different from y'_i and $c_i(x) = 0$ happens with probability $\beta(1 - \kappa)$ (on average).

Then, the value of the parameter β must be chosen in accordance with the value of κ (the value κ is fixed by choosing the polynomials c_i, $1 \leq i \leq m$).

3.3 Relation between the parameters α, β and κ

Recall that α is the number of equation of the public key that must be satisfied to valid an authentication or a signature. The parameters β and κ concerns the two

perturbations involved in a multivariate probabilistic scheme based on a known trapdoor: the value β is the probability that a bit of the received challenge y is modified by the prover before inverting the system, and the value κ is the probability that a polynomial equation of the perturbation system $\mathcal{C}$ equals 1.

The value α depends on the probability that the equation $y_i \stackrel{?}{=} a_i(x_1, \ldots, x_n)$, $1 \leq i \leq m$, is not satisfied, that is, α depends on the two values β and κ. There are (on average) κm integers $i \in [1; m]$ such that $y'_i = b_i(x_1, \ldots, x_n) + c_i(x_1, \ldots, x_n)$ and the prover has changed βm values of y. Thus, the parameter α must be chosen such that:

$$\alpha \simeq (\kappa - \beta)m \ .$$

Since the probability κ is fixed by choosing the polynomials c_i, $1 \leq i \leq m$, the values of α and β must be chosen in accordance with the value of κ. Notice that we must choose α such that $\alpha > \frac{m}{2}$ in order to prevent that a random value is valid with a probability $\frac{1}{2}$ and β must be different from 0.

3.4 Size of the public key

Let $K = GF(2)$ and assume that the equations of the public key look as random equations of degree d for an adversary who do not have the secret key. We have $\frac{m}{2} < \alpha \leq m$. Let λ be the value defined by $\alpha = \lambda m$. Then, in order to ensure a security in 2^{80}, the number m of equations of a public key must be chosen such that:

$$m \left(1 + \lambda \frac{\ln \lambda}{\ln 2} + (1 - \lambda) \frac{\ln(1 - \lambda)}{\ln 2} \right) \simeq 80 \ .$$

Details of this approximation are given in Appendix A.

Example 1. For $\lambda = \frac{3}{4}$, we get $m \simeq 423$, and for $\lambda = \frac{9}{10}$, we get $m \simeq 150$ equations.

Remark 5. As a consequence, the public key is larger in a probabilistic scheme than in a non-probabilistic where at least about 80 equations are required.

4 The probabilistic multivariate scheme $C^* + LL'$

The Matsumoto-Imai scheme (also called C^*) was presented in [13] and crypt-analysed in [14, 3]. We first briefly recall the description of C^*. Next, we present a probabilistic variation of the C^* scheme, called $C^* + LL'$ where no attack is known; another way to repair the C^* scheme is for example the FLASH scheme of [18].

In this section, the public key $\mathcal{A} = \mathcal{B} + \mathcal{C}$ will be constructed such that $\mathcal{B}$ is a public key of a C^* scheme and $\mathcal{C}$ is a set of product of linear forms ($\mathcal{B}$ and $\mathcal{C}$ are kept secret).

4.1 Matsumoto-Imai Scheme (C^*)

Let $K = \mathbb{F}_q$ be a finite field and $\mathbb{E}$ be an extension field of dimension n over K. Let Φ be an isomorphism from $\mathbb{E}$ to K^n. Let f be the function defined over $\mathbb{E}$ by

$$f : x \longmapsto x^{1+q^\theta},$$

where $\theta \in \mathbb{N}$. If the finite field K has characteristic 2 and $\gcd(q^n - 1, q^\theta + 1) = 1$, then f is a bijection. Furthermore, the restriction on θ allows an efficient inversion of the function f. Indeed, $f^{-1}(y) = y^{h'}$, where h' is the inverse of $1 + q^\theta$ modulo $q^n - 1$.

The public key is the function $A := x \mapsto T \circ \Phi \circ f \circ \Phi \circ S(x)$. The hardness of the Matsumoto-Imai scheme is based on the IP-problem, that is, the difficulty of finding transformations S and T for given polynomials equations P and P'.

4.2 Construction of the public key $\mathcal{A}$

Let $K = GF(2)$. Let $\mathcal{B}$ be the public key of a C^* scheme, that is, $\mathcal{B}$ is a set of n quadratic equations in n variables over $GF(2)$ of the form

$$y_i = b_i(x_1, \ldots, x_n)$$

with $1 \leq i \leq n$ and $x_1, \ldots, x_n$, $y_1 \ldots, y_n$ are elements of K. The trapdoor associated to $\mathcal{B}$ is denoted by $T_{\mathcal{B}}$. Notice that both $\mathcal{B}$ and $T_{\mathcal{B}}$ are kept secret.

Let $L_1, \ldots, L_n, L'_1, \ldots, L'_n$ be $2n$ secret linear forms in the variables $x_1, \ldots, x_n$. For every i, $1 \leq i \leq n$, let $c_i = L_i \cdot L'_i$. Then, the public key $\mathcal{A}$ of the scheme $C^* + LL'$ is the set of the n quadratic equations in n variables:

$$\mathcal{A} = \begin{cases} y_1 &= b_1(x_1, \ldots, x_n) + L_1(x_1, \ldots, x_n) \cdot L'_1(x_1, \ldots, x_n) = a_1(x_1, \ldots, x_n) \\ &\vdots \\ y_n &= b_n(x_1, \ldots, x_n) + L_n(x_1, \ldots, x_n) \cdot L'_n(x_1, \ldots, x_n) = a_n(x_1, \ldots, x_n) \end{cases}$$

Remark 6. The classification of quadratic forms over $GF(q)$ (for q odd or even) is well-known; it is given for example in [11] pp. 278-289 and recalled in Appendix B. We are interested here in the case q even since q is generally a power of two. Then, we have only one or two canonic forms when n is fixed and non degenerate, so we have at least $2n$ possible canonic forms when q is fixed.

Parameter κ. For all i, $1 \leq i \leq n$, we have $L_i(x_1, \ldots, x_n) = 0$ with a probability $\frac{1}{2}$ and we also have $L'_i(x_1, \ldots, x_n) = 0$ with a probability $\frac{1}{2}$. Thus, we have $L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n) = 0$ with a probability $\kappa = \frac{3}{4}$.

Recall that the parameter α is a fixed value such that:

$$\alpha \simeq (\kappa - \beta)\, m = \left(\frac{3}{4} - \beta\right) m \ ,$$

where β is a fixed parameter such that $0 < \beta < \frac{1}{4}$.

4.3 The scheme $C^* + LL'$

As usual, $y = (y_1, \ldots, y_n)$ is the challenge of an authentication scheme or the hash value of the message to be signed in a signature scheme. The value $x = (x_1, \ldots, x_n)$ will be a successful authentication value or a valid signature if at least α equations of $\mathcal{A}$ are satisfied.

Computation of the value x. Let $y = (y_1, \ldots, y_n)$ be the challenge or the hash value of the message to be signed. In order to compute $x = (x_1, \ldots, x_n)$, the prover proceeds as follows.

1. Each y_i is changed with probability β and then the value $y' = (y'_1, \ldots, y'_n)$ is obtained. The indices i such that $y_i \neq y'_i$ are chosen with a deterministic pseudo-random algorithm that depends only of $(y_1, \ldots, y_n)$ and of the current run. Thus, if the same challenge is given twice, the prover will always answer with the same $(x_1, \ldots, x_n)$.
2. Using the trapdoor $T_{\mathcal{B}}$, the prover computes the value $x = (x_1, \ldots, x_n)$ such that:
$$\forall i, \quad 1 \leq i \leq n, \quad y'_i = b_i(x_1, \ldots, x_n) \ .$$

Verification of the value x. The value x is valid if at least α equations of the public key, i.e. if at least α equations of the form

$$y_i \stackrel{?}{=} a(x_1, \ldots, x_n)$$

where $1 \leq i \leq m$, are valid.

For every $i, 1 \leq i \leq m$, we have $L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n) = 0$ with probability $\kappa = \frac{3}{4}$. Then, we have $y'_i = b_i(x_1, \ldots, x_n) + L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n)$ with a probability $\frac{3}{4}$. Next, we have $y_i = y'_i$ with a probability $(1 - \beta)$. Thus, we deduce that we have:

$$y_i = f_i(x_1, \ldots, x_n) + L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n)$$

with a probability greater than or equal to $\frac{3}{4} - \beta$.

Then, the expectation value of the number N of equations of $\mathcal{A}$ that are satisfied is greater than or equal to $\left(\frac{3}{4} - \beta\right) n \simeq \alpha$. For a given $(y_1, \ldots, y_n)$, if N is lower than α, then we can try again at step 1 by computing another $(y'_1, \ldots, y'_n)$ with again about βn values changed from $(y_1, \ldots, y_n)$ chosen with a deterministic pseudo-random algorithm that depends only of $(y_1, \ldots, y_n)$ and of the current run. After a few tries, we get a solution $(x_1, \ldots, x_n)$ with at least α equations of $\mathcal{B}$ that are satisfied, i.e., a valid signature or a valid answer to a challenge.

Remark 7. For a security greater than or equal to 2^{80}, we need $n \geq 423$ when β is small. For instance, with $\beta = \frac{1}{10}$ and $n \simeq 500$, no attack of this scheme exists to the best of our knowledge.

4.4 Variants of the scheme $C^* + LL'$

First variant: $C^* + LL' + L''L'''$. The first variant consists in replacing the linear product LL' by the linear product $LL' + L''L'''$ (as a consequence, the value of the parameter κ is modified). We keep the same notations, that is, $\mathcal{B}$ is a public key of a C^* scheme and $\mathcal{A}$ is the set of n equations of the form:

$$y_i = b_i(x_1, \ldots, x_n) + c_i(x_1, \ldots, x_n) = a_i(x_1, \ldots, x_n) \ ,$$

where c_i, $1 \leq i \leq n$, is a product a linear forms which is defines as follows.

Let L_i, L'_i, L''_i and L'''_i, $1 \leq i \leq n$, be $4n$ secret linear forms in the n variables $x_1, \ldots, x_n$. The set $\mathcal{C}$ is defined by:

$$\mathcal{C} = \begin{cases} y_1 = & b_1(x_1, \ldots, x_n) + L_1(x_1, \ldots, x_n)L'_1(x_1, \ldots, x_n) \\ & +L''_1(x_1, \ldots, x_n)L'''_1(x_1, \ldots, x_n) \\ \quad \vdots & \\ y_n = & b_n(x_1, \ldots, x_n) + L_n(x_1, \ldots, x_n)L'_n(x_1, \ldots, x_n) \\ & +L''_n(x_1, \ldots, x_n)L'''_n(x_1, \ldots, x_n) \end{cases}$$

The value of the parameter κ is the probability that the equation $L_i \cdot L'_i + L''_i \cdot L'''_i \stackrel{?}{=} 0$ is satisfied, that is, $\kappa = \frac{10}{16}$ according to the figure 2.

L_i	L'_i	L''_i	L'''_i	$L_i \cdot L'_i + L''_i \cdot L'''_i$
1	1	1	1	0
1	1	1	0	1
1	1	0	1	1
1	1	0	0	1
1	0	1	1	1
1	0	1	0	0
1	0	0	1	0
1	0	0	0	0
0	1	1	1	1
0	1	1	0	0
0	1	0	1	0
0	1	0	0	0
0	0	1	1	1
0	0	1	0	0
0	0	0	1	0
0	0	0	0	0

Fig. 2. Truth table of $L_iL'_i + L''_iL'''_i$

Since we have $\frac{1}{2} < \kappa = \frac{10}{16} < \frac{3}{4}$, the scheme $C^* + LL' + L''L'''$ will generally be less efficient than the scheme $C^* + LL'$. However, it may be difficult to distinguish the public key of $C^* + LL' + L''L'''$ from random quadratic equations than the

public key of C^*+LL', and thus, for C^* public key $\mathcal{B}$, the scheme $C^*+LL'+L''L'''$ may be more secure than the scheme $C^* + LL'$.

More generally, from [11], we know what can be the exact numbers of solutions $x_1, \ldots, x_n$ of any quadratic form $q(x_1, \ldots, x_n) = 0$. For instance, the number of $(x_1, \ldots, x_n) \in \mathbb{F}_2^n$ such that $x_1x_2 + x_3x_4 + \cdots + x_{n-1}x_n = 0$ with n even is $2^{n-1} + 2^{\frac{n-2}{2}}$, i.e.:

$$2^{n-1}\left(1 + \frac{1}{2^{\frac{n}{2}}}\right)$$

instead of 2^{n-1} for an average quadratic form of n variables.

Second variant: decomposing $\mathcal{A}$ in sets of equations with various probability. Instead of having about 423 equations $C^* + LL'$ in $\mathcal{A}$, we can for example have 40 equations that come from a C^{*--} scheme (these equations will have to be satisfied with a probability 100%) and 160 equations that come from a $C^* + LL'$ scheme and at least 120 of these equations will have to be satisfied. Many other examples are possible for the parameters.

Third variant: public key of degree 3 instead of 2 When using a public key formed with quadratic polynomials, it is not possible to prevent the attacker that observe an equation $y_i \neq a(x)$ from distinguishing between the two cases :

1. $y_i = y'_i$ and $L_i(x) \cdot L'_i(x) = 1$
2. $y_i \neq y'_i$ and $L_i(x) \cdot L'_i(x) = 0$

Indeed, we have $y_i = y'_i$ with probability $(1-\beta)$ and we have $L_i(x) \cdot L'_i(x) = 0$ with probability κ. Then, in order to prevent the attacker from distinguishing between case 1 and case 2, we have to choose the values of β and κ such that:

$$(1 - \beta)(1 - \kappa) = \beta\kappa$$

Furthermore, we have $\alpha \approx (\kappa - \beta)n \geq \frac{m}{2}$. That comes to choose the values of κ and β such that: $\kappa + \beta = 1$ and $\kappa - \beta > \frac{1}{2}$.

These conditions imply that $\kappa > \frac{3}{4}$. When the public key has degree 2 then, the higher value of κ is $\frac{3}{4}$ (c.f. the weight distribution of quadratic forms). If $\kappa = \frac{3}{4}$, then there is no solution β fulfilling both $\kappa + \beta = 1$ and $\kappa - \beta > \frac{1}{2}$.

This property can be achieved by using public key of degree 3. In a C^* scheme, a monomial $b = a^{1+q^\theta}$ is hidden by affine transformations. In [15], the possibility of replacing $b = a^{1+q^\theta}$ by $b = a^{1+q^\theta+q^\varphi}$ is studied; the public key has degree 3 instead of 2. The attack of the scheme C^* given in [14] does not apply directly on the scheme "C^* of degree 3". However the scheme is insecure as it is shown in [15]. We use the scheme "C^* of degree 3" as a basic scheme to construct a probabilistic multivariate scheme.

Let $\mathcal{B}$ be the public key of a scheme "C^* of degree 3", that is $\mathcal{B}$ is a set of n equations in n variables of degree 3 over K of the form $y_i = b_i(x_1, \ldots, x_n)$ where

$1 \leq i \leq n$ and $x_1, \ldots, x_n, y_1, \ldots, y_n$ are elements of K. The trapdoor associated to $\mathcal{B}$ is denoted by $T_{\mathcal{B}}$.

Let $L_1, \ldots, L_n, L'_1, \ldots, L'_n, L''_1, \ldots, L''_n$ be $3n$ secret linear forms in the variables $x_1, \ldots, x_n$. Then, the public key $\mathcal{A}$ is the set of the n equations of degree 3 in n variables:

$$\mathcal{A} = \begin{cases} y_1 &= b_1(x_1, \ldots, x_n) + L_1(x_1, \ldots, x_n)L'_1(x_1, \ldots, x_n)L'_1(x_1, \ldots, x_n) \\ &\vdots \\ y_m &= b_1(x_1, \ldots, x_n) + L_m(x_1, \ldots, x_n)L'_m(x_1, \ldots, x_n)L'_m(x_1, \ldots, x_n) \end{cases}$$

Parameter κ. For all i, $1 \leq i \leq n$, we have $L_i(x_1, \ldots, x_n) = 0$ with a probability $\frac{1}{2}$ and we also have $L'_i(x_1, \ldots, x_n) = 0$ and $L''_i(x_1, \ldots, x_n) = 0$ with a probability $\frac{1}{2}$. Thus, we have $L_1(x_1, \ldots, x_n)L'_1(x_1, \ldots, x_n)L'_1(x_1, \ldots, x_n) = 0$ with probability $\kappa = \frac{7}{8}$.

Parameters α and β. Recall that the three parameters α, β and κ must satisfy the relation:

$$\alpha \simeq (\kappa - \alpha)n = \frac{3}{4}n \geq \frac{n}{2} \ .$$

By choosing $\beta = 1 - \kappa = \frac{1}{8}$, an attacker would not be able to distinguish between the two possible cases when a relation of the public key is not satisfied.

5 The probabilistic multivariate scheme *UOV + LL'*

The scheme *Oil and Vinegar* was introduced in [17] and it was broken in [9]. Next, a generalisation of the original scheme, called *Unbalanced Oil and Vinegar (UOV)*, was introduced in [8]; the scheme UOV is not broken for well-chosen parameters. In this section, we will be able to use more possible parameters since some attacks valid for UOV will not work any more for UOV+LL'. We briefly describe the scheme *UOV*.

5.1 The scheme UOV

Let $K = \mathbb{F}_q$ be a small finite field. Let m, n and p be three positive integers. The hash value y of the message to be signed is an element of K^m, and the signature x is an element of K^n.
The public key is a set $\mathcal{A}$ of m polynomials in n variables of the form:

$$y_i = f_i(x_1, \ldots, x_n), \quad 1 \leq i \leq m \ .$$

There exists a bijective affine function $s : K^n \to K^n$ such that:

$$(x_1, \ldots, x_n) = s(o_1, \ldots, o_{n-p}, v_1, \ldots, v_p)$$

and such that for every i, $1 \leq i \leq m$:

$$y_i = \sum_{j=1}^{n-p} \sum_{k=1}^{p} \gamma_{i,j,k} o_j v_k + \sum_{j=1}^{p} \sum_{k=1}^{p} \mu_{i,j,k} v_j v_k + \sum_{j=1}^{n-p} \delta_{i,j} o_j + \sum_{j=1}^{p} \nu_{i,j} b_j + \xi_i$$

Note that the vinegar variables v_i's are combined quadratically while the oil variables o_i's are only combined with vinegar variables in a quadratic way. Therefore assigning random values to the vinegar variables results in a system of linear equations in the oil variables which can be solved, for instance, by using gaussian elimination.

The scheme UOV+LL' proceeds exactly as the scheme $C^* + LL'$ except that the C^* equations are changed with UOV equations. Since this UOV+LL' scheme looks particularly interesting, we will describe it completely in this section and give some remarks on its efficiency.

5.2 Construction of the public key $\mathcal{A}$

Let $K = GF(2)$ and $\mathcal{B}$ be the public key of a UOV scheme, i.e., $\mathcal{B}$ is a set of m quadratic equations in n variables $(x_1, \ldots, x_n)$ over $GF(2)$. Each equation of $\mathcal{B}$ is of the form:

$$y_i = f_i(x_1, \ldots, x_n)$$

with $x_1, \ldots, x_n, y_i \in K$, and f_i is a quadratic function.

There are $n - p$ *oil variables* denoted by $o_1, \ldots, o_{n-p} \in K$ and p *vinegar variables* denoted by $v_1, \ldots, v_p \in K$ and there is a secret affine and invertible transformation s such that:

$$(x_1, \ldots, x_n) = s(o_1, \ldots, o_{n-p}, v_1, \ldots, v_p)$$

and such that each y_i of $\mathcal{B}$ written in the $o_1, \ldots, o_{n-p}, v_1, \ldots, v_p$ variables (instead of $x_1, \ldots, x_n$ variables) is of the form:

$$y_i = \sum_{j=1}^{n-p} \sum_{k=1}^{p} \gamma_{i,j,k} o_j v_k + \sum_{j=1}^{p} \sum_{k=1}^{p} \mu_{i,j,k} v_j v_k + \sum_{j=1}^{n-p} \delta_{i,j} o_j + \sum_{j=1}^{p} \nu_{i,j} v_j + \xi_i$$

where $\gamma_{i,j,k}$, $\mu_{i,j,k}$, $\delta_{i,j}$, $\nu_{i,j}$ and ξ_i are elements of K. Notice that we do not have any term in $a_i a_j$: we can have oil $\times$ vinegar, vinegar $\times$ vinegar but never oil $\times$ oil.

Let $L_1, \ldots, L_m, L'_1, \ldots; L'_m$ be $2m$ secret linear forms in $x_1, \ldots, x_n$ (or equivalently in the variables $a_1, \ldots, a_h, b_1, \ldots, b_v$). Let $\mathcal{A}$ be the set of the m quadratic equations:

$$y_i = f_i(x_1, \ldots, x_n) + L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n) \ .$$

The set $\mathcal{A}$ will be the public key of the scheme UOV+LL' (while f_i, $\mathcal{B}$, L_i, L'_i and s are kept secret).

5.3 The scheme $UOV + LL'$

As usual $y = (y_1, \ldots, y_m)$ is the challenge in an authentication scheme, or the hash value of the message to be signed in a signature scheme. The value $x = (x_1, \ldots, x_n)$ is a valid signature or a successful authentication if at least α equations of $\mathcal{A}$ are satisfied, with $\alpha \simeq \left(\frac{3}{4} - \beta\right) m$, where β is a fixed parameter (for example, we can choose $\beta \simeq \frac{1}{10}$).

Computation of the value x. In order to compute $x = (x_1, \ldots, x_n)$ with the secrets, the prover proceeds as follows.

1. y_i is changed with probability β and then the value $y' = (y'_1, \ldots, y'_m)$ is obtained. The indices i such that $y_i \neq y'_i$ are chosen with a deterministic pseudo-random algorithm that depends only of $(y_1, \ldots, y_m)$. Thus, if the same challenge is given twice, the prover will always answer with the same $(x_1, \ldots, x_n)$.
2. The prover randomly chooses the *vinegar variables* $v_1, \ldots, v_p$. These random variables are chosen with a deterministic pseudo-random algorithm that depend only of $(y_1, \ldots, y_m)$. Thus, as above, if twice the same challenge $(y_1, \ldots, y_m)$ is given, the prover will always answer with the same $(x_1, \ldots, x_n)$.
3. The prover computes the values $a_1, \ldots, a_m$ such that:

$$\forall i, 1 \leq i \leq m, y'_i = f_i(x_1, \ldots, x_n) = f_i(s(o_1, \ldots, o_{n-p}, v_1, \ldots, v_p))$$

   Here we have a linear system of m equations in the variables $o_1, \ldots, o_{n-p}$. If we have no solution we try again with other random vinegar values $v_1, \ldots, v_p$.

For all i, $1 \leq i \leq m$, we have:

$$y'_i = f_i(x_1, \ldots, x_n) + L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n) \ .$$

with a probability $\frac{3}{4}$. Moreover, with a probability $(1 - \beta)$, we have $y_i = y'_i$. Thus, with a probability greater than or equal to $\frac{3}{4} - \beta$ we have:

$$y_i = f_i(x_1, \ldots, x_n) + L_i(x_1, \ldots, x_n) \cdot L'_i(x_1, \ldots, x_n).$$

Then, the expectation value of the number N of equations of $\mathcal{A}$ that are satisfied is greater than or equal to $\left(\frac{3}{4} - \beta\right) \simeq \alpha$. If we have $N < \alpha$, then we can try again with new random vinegar variables.

Remark 8. If we compare UOV and UOV+LL', we can notice that in UOV+LL' we do not need any more to have $v \geq 2m$ in order to avoid the Shamir-Kipnis attack of [9]. Moreover in the equations of UOV, we have oil $\times$ oil, oil $\times$ vinegar and vinegar $\times$ vinegar, so the scheme might be more secure for smaller values of the parameters.

Remark 9. The variations given in Section 3.2 and 3.3 for $C^* + LL'$ are also possible variants for $UOV + LL'$.

6 Conclusion

Probabilistic Multivariate Cryptography is a new concept in public key cryptography with many possible schemes. It opens new opportunities and new questions that we think are interesting, both from a practical and from a theoretical point of view. In this paper we have presented some new public key schemes (C* + L L' and UOV + LL' for example) based on this idea of probabilistic Multivariate Cryptography with some explicit examples for the parameters. These schemes were built from the transformation of non-probabilistic multivariate schemes to probabilistic multivariate schemes in order to get more security or more efficiency. An interesting problem is to find a trapdoor for probabilistic multivariate schemes which allows directly to find an approximation of the solution associated to the challenge or the message to be signed. Another interesting problem is to find probabilistic multivariate schemes for encryption (not only for signatures or authentications).

Acknowledgements

The authors wish to thank the anonymous referees for useful comments.

References

1. Julien Bringer, Herv Chabanne, and Emmanuelle Dottax. Perturbing and protecting a traceable block cipher. http://eprint.iacr.org/, 2006.
2. N. Courtois. The security of hidden field equations (HFE). In *Progress in Cryptology - CT-RSA 2001: The Cryptographers' Track at RSA Conference 2001*, volume 2020, pages 266–281, San Francisco, CA, USA, 2001.
3. P. Delsarte, Y. Desmedt, A.M. Odlyzko, and P. Piret. Fast cryptanalysis of the Matsumoto-Imai public key scheme. In *Advances in Cryptology: Proceedings of Eurocrypt'84 - A workshop on the Theory and Application of Cryptographic Techniques*, volume 209, pages 142–149, Paris, France, 1984.
4. J. Ding. A new variant of the Matsumoto-Imai cryptosystem through perturbation. In *Public Key Cryptography PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2947, pages 305–318, Singapore, 2004.
5. H. Fell and W. Diffie. Analysis of a public key approach based on polynomial substitution. In *Lecture notes in computer sciences on Advances in Cryptology – Crypto'85*, volume 218, pages 340–349. Springer-Verlag, 1986.
6. P-A. Fouque, L. Granboulan, and J. Stern. Differential cryptanalysis for multivariate schemes. In *Advances in Cryptology EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, volume 3494, pages 341–353, Aarhus, Denmark, 2005.
7. H. Imai and T. Matsumoto. Algebraic methods for constructing asymetric cryptosystems. In *Proceedings of Algebraic Algorithms and Error-Correctings Codes – AAECC*, pages 108–119. Lecture Note in Computer Science, Springer-Verlag, 1985.
8. A. Kipnis, J. Patarin, and L. Goubin. Unbalanced oil and vinegar signature schemes. In *Lecture Notes in Computer Science on Advances in Cryptology – Eurocrypt'99*, volume 1592, pages 206–222. Springer-Verlag, 1999.

9. A. Kipnis and A. Shamir. Cryptanalysis of the oil & vinegar signature scheme. In *Lecture Notes in Computer Science on Advances in Cryptology – Crypto'98*, volume 1462, pages 257–266, London, UK, 1998. Springer-Verlag.
10. A. Kipnis and A. Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In *Lecture Notes in Computer Science on Advances in Cryptology – Crypto'99*, volume 1666, pages 19–30. Springer-Verlag, 1999.
11. R. Lidl and H. Niederreiter. *Finite fields*, volume 20 of *Encyclopedia of Mathematics and its applications*. Cambridge University Press, New York, NY, USA, 1997.
12. F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*. Elsevier, North-Holl., 1977.
13. T. Matsumoto and H. Imai. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In *Lecture Notes in Computer Science on Advances in Cryptology – Eurocrypt'88*, volume 330, pages 419–453. Springer-Verlag, 1988.
14. J. Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88. In *Lecture Notes in Computer Science on Advances in Cryptology – Crypto'95*, volume 963, pages 248–261, Santa Barbara, California, USA, 1995. Springer-Verlag.
15. J. Patarin. Asymmetric cryptography with a hidden monomial. In *Lecture Notes in Computer Science on Advances in Cryptology – Crypto'96*, volume 1109, pages 45–60, Santa Barbara, California, USA, 1996. Springer-Verlag.
16. J. Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two new families of asymmetric algorithms. In *Lecture Notes in Computer Science on Advances in Cryptology - EUROCRYPT'96*, volume 1070, pages 33–48, Saragossa, Spain, 1996.
17. J. Patarin. The oil and vinegar signature scheme. Presented at the Dagstuhl Workshop on Cryptography, 1997.
18. J. Patarin, N. Courtois, and L. Goubin. Flash, a fast multivariate signature algorithm. In *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, pages 298–307, London, UK, 2001. Springer-Verlag.
19. J. Patarin, N. Courtois, and L. Goubin. Quartz, 128-bit long digital signatures. In *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, pages 282–297, London, UK, 2001. Springer-Verlag.
20. J. Patarin, L. Goubin, and N. Courtois. C^*_{-+} and HM: Variations around two schemes of T. Matsumoto and H. Imai. In *Advances in Cryptology - ASIACRYPT'98: International Conference on the Theory and Application of Cryptology and Information Security*, volume 1514, pages 35–49, Beijing, China, 1998.
21. A. Shamir. Efficient signature schemes based on birational permutations. In *Advances in Cryptology - CRYPTO '93: 13th Annual International Cryptology Conference*, volume 773, pages 1–12, Santa Barbara, California, USA, 1994.
22. C. Wolf, A. Braeken, and B. Preneel. Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC. In *In Conference on Security in Communication Networks – SCN 2004, Lecture notes in computer sciences*, volume 3352, pages 145–151.
23. C. Wolf and B. Preneel. Taxonomy of public key schemes based on the problem of multivariate quadratic equations. this paper is available at `http://www.esat.kuleuven.ac.be/cosic/`.

A Size of the public key

Let $K = GF(2)$. We want to evaluate the minimum number of equations of a public key in order to ensure a security in 2^{80}. Notice that, we also assume that the equations look as random equations of degree d for an adversary who do not have the secret key.

Given a hash value of a message or a challenge $y \in K^m$, an adversary can choose a random value $x \in K^n$ for the signature or the authentication value. For each try, the attacker has a probability $\frac{1}{2^m} \sum_{i=\alpha}^{m} \binom{m}{i}$ to have α or more satisfied equations. Then, m must be chosen such that:

$$\frac{1}{2^m} \sum_{i=\alpha}^{m} \binom{m}{i} \leq \frac{1}{2^{80}} \ .$$

We have $\frac{m}{2} < \alpha \leq m$. Let λ be the value defined by $\alpha = \lambda m$. If λ is sufficiently different from $\frac{1}{2}$, then the dominant term in $\sum_{i=\alpha}^{m} \binom{m}{i}$ is $\binom{m}{\alpha}$. More precisely, we can overvalue $\sum_{i=\alpha}^{m} \binom{m}{i}$ by a geometric sum with the first term $\binom{m}{\alpha}$. Thus, we want to evaluate:

$$\frac{1}{2^m} \binom{m}{\alpha} = \frac{1}{2^m} \cdot \frac{m!}{\alpha!(m-\alpha)!} = \frac{1}{2^m} \frac{m!}{(\lambda m)!\,(m(1-\lambda))!} \ .$$

From stirling formula $n! \sim n^n \exp^{-n} \sqrt{2\pi n}$, we get:

$$\frac{1}{2^m} \binom{m}{\alpha} \approx \frac{1}{2^m} \frac{m^m \exp^{-m} \sqrt{2\pi m}}{(\lambda m)^{\lambda m} \exp^{-\lambda m} \sqrt{2\pi\lambda m} \cdot (m(1-\lambda))^{m(1-\lambda)} \exp^{-m(1-\lambda)} \sqrt{2\pi m(1-\lambda)}} \ .$$

After simplifications, we get:

$$\frac{1}{2^m} \binom{m}{\alpha} \approx \frac{1}{2^m \left(1 + \lambda \frac{\ln \lambda}{\ln 2} + (1-\lambda) \frac{\ln(1-\lambda)}{\ln 2}\right) \sqrt{2\pi m\lambda(1-\lambda)}} \ .$$

In first approximation, this will be about $\frac{1}{2^{80}}$ when $m \left(1 + \lambda \frac{\ln \lambda}{\ln 2} + (1-\lambda) \frac{\ln(1-\lambda)}{\ln 2}\right) \simeq 80$.

B Classification of quadratic forms over $GF(q)$

The classification of quadratic forms over $GF(q)$ (for q odd or even) is well-known; it is given for example in [11] pp. 278-289. We are interested here in the case q even since q is generally a power of two. Then, we recall here the two main theorems for the case q even.

Theorem 1 ([11] p.287). *Let $GF(q)$ be a finite field with q even. Let $f \in GF(q)[x_1, \ldots, x_n]$ be a non degenerate quadratic form. If n is odd, then f is equivalent to:*

$$x_1x_2 + x_3x_4 + \ldots, x_{n-2}x_{n-1} + x_n^2 \ .$$

If n is even, then f is equivalent to one of the two forms:

1. $x_1x_2 + x_3x_4 + \ldots, x_{n-1}x_n$
2. $x_1x_2 + x_3x_4 + \ldots, x_{n-1}x_n + x_{n-1}^2 + ax_n^2$

where $a \in GF(q)$ *satisfies* $Tr_{GF(q)}(a) = 1$.

Theorem 2 ([11] p.288). *Let* $GF(q)$ *be a finite field with* q *even. Let* $b \in GF(q)$.
For odd n*, the number of solutions of the equation*

$$x_1x_2 + x_3x_4 + ... + x_{n-2}x_{n-1} + x_n^2 = b$$

in $GF(q)^n$ *is* q^{n-1}.
For even n*, the number of solutions of the equation*

$$x_1x_2 + x_3x_4 + ... + x_{n-1}x_n = b$$

in $GF(q)^n$ *is* $q^{n-1} + \nu(b)q^{\frac{n-2}{2}}$*, with* $\nu(b) = -1$ *if* $b \neq 0$ *and* $\nu(0) = q - 1$.
For even n *and* $a \in GF(q)$ *with* $Tr_{GF(q)}(a) = 1$*, the number of solutions of the equation*

$$x_1x_2 + x_3x_4 + ... + x_{n-1}x_n + x_{n-1}^2 + ax_n^2 = b$$

in $GF(q)^n$ *is* $q^{n-1} - \nu(b)q^{\frac{n-2}{2}}$*, with* $\nu(b) = -1$ *if* $b \neq 0$ *and* $\nu(0) = q - 1$.

Then, we have only one or two canonic forms when n is fixed and non-degenerate, so we have at most $2n$ possible canonic forms when q is fixed. This number is generally too small to give any useful information in our schemes, for example when the transformation LL' is applied.

ℓ-Invertible Cycles for $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Public Key Cryptography

Jintai Ding[1], Christopher Wolf[2], and Bo-Yin Yang[3]
[1]ding@math.uc.edu, University of Cincinnati, Cincinnati, Ohio, USA.
[2]Christopher.Wolf@ens.fr,chris@Christopher-Wolf.de,
École Normale Supérieure, Paris, France.
[3]by@moscito.org, Tamkang University *and* Taiwan Info. Security Center.

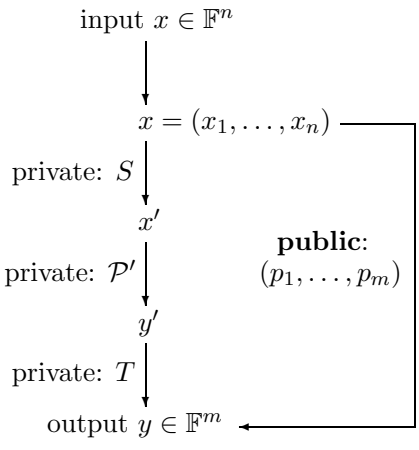
Abstract

We propose a new basic trapdoor ℓIC (ℓ-Invertible Cycles) for $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key cryptosystems of the mixed field type. While ℓIC is distantly related to the well-known C^* or Matsumoto-Imai Scheme A (MIA) trapdoor, and share some features of the stagewise triangular systems, it has distinctive properties that sets it apart as a new basic trapdoor in the context of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key systems. This is the first new basic trapdoor since the invention of Unbalanced Oil and Vinegar in 1997, nearly a decade ago. In practice, ℓIC is much faster than MIA, and can even match the speed of single-field $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic schemes. We also introduce formally a new modifier, the so-called "embedding", which can be used to construct mixed field schemes without decryption failure.

Keywords: Public Key, $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic, Basic Trapdoor, Encryption, Signing

1 Introduction

We discuss $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic ($\mathcal{MQ}$) public key cryptosystems. These first appeared in the English literature in the mid '80s [FD85, IM85] as alternatives to RSA and other traditional PKCs. Often it is remarked that we should maintain active research into alternative asymmetric crypto algorithms for ecological diversity. Quite true when Quantum Computers and poly-time cracking of factoring and discrete-log-based PKCs (Shor's algorithm [Sho97]) creeps closer to practicality. We also hope to show that $\mathcal{MQ}$ PKCs are independently interesting on their own merits.



The finite field $\mathbb{F}$ of q elements is called the *base field*. $\mathcal{P}' \in \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$ is a system of m quadratic polynomials in n variables in $\mathbb{F}$, called the *central map* and its components *central polynomials*. Composition with the affine maps S, T masks the structure of $\mathcal{P}'$ and gives the public map:

$$\mathcal{P} = (p_1, \ldots, p_m) \quad := \quad T \circ \mathcal{P}' \circ S \tag{1}$$

We usually write, for $1 \le i \le m, 1 \le j \le k \le n$,

$$p_i(x_1, \ldots, x_n) := \sum_{1 \le j \le k \le n} \gamma_{i,j,k} x_j x_k + \sum_{j=1}^{n} \beta_{i,j} x_j + \alpha_i$$

where α_i is usually normalized to zero. The $mn(n+3)/2$ coefficients $\gamma_{ijk}, \beta_{ij} \in \mathbb{F}$ form the public key.

Figure 1: Illustration of Terminology and Notation for an modern $\mathcal{MQ}$-trapdoor

We need to invert $\mathcal{P}'$ efficiently for PKCs. A simple method to build $\mathcal{P}'$ for consequent inversion is a *basic trapdoor*. Trapdoors can be combined or modified slightly to create variants. "Modifiers" are systemic ways to derive new trapdoors from old (cf. Sec. 4, [WP05b, Sec. 4]).

There are four previously known basic trapdoors (in the terminology of [WP05b]):

Mixed-Field (or "Big Field"): Operates mostly over a much larger extension field $\mathbb{E} = \mathbb{F}^k$.

MIA: Matsumoto-Imai Scheme A or C^* ([IM85], Imai-Matsumoto, '85).

HFE: Hidden Field Equations ([Pat96], Patarin, '96), a generalization of MIA.

Single-Field (or "True"): Works for the most part on the individual components of x' and y'.

UOV: Unbalanced Oil and Vinegar ([Pat97, KPG99], Patarin *et al*, '97 onwards).

STS: Stepwise Triangular System (in Japanese – lectures seen in 1985 – [TKI+86], Tsujii); [Sha93], Shamir 1993). Generalized later to its present form [GC00, WBP04].

Recent combination trapdoors MFE/TRMC or enTTS/TRMS/Rainbow [DS05b, WHL+05, WYHL06, YC05] are built from triangular stages — in layers or coupled with UOV.

Outline. In the next section, we introduce our new trapdoor and discuss its basic properties. In particular, we show that certain instances can be inverted very quickly. Section 3 give cryptanalytic properties of this basic trapdoor and enumerates possible attacks. Section 4 discusses countermeasures to these attacks, *i.e.*, *modifiers*. We give the practical instances in Section 5. These we verify to withstand known attacks. The main text of the paper concludes with Section 6.

We list some pertinent background and results in the appendices. Appendix A discusses possible extensions of ℓIC. Appendix C investigates the space of linearizing equations. Appendix B discusses the state of the art for equation solvers and how they relate to ℓIC. The last appendix discuss how to find parameters for secure signature schemes.

2 ℓ-Invertible Cycles (ℓIC)

In this section, we will introduce a new basic trapdoor for $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic ($\mathcal{MQ}$) public key cryptography. We want to stress that this trapdoor does not fit into the taxonomy developed in [WP05b], and hence, we have a fifth basic trapdoor, next to MIA (Matsumoto-Imai Scheme A), HFE (Hidden Field Equations), UOV (Unbalanced Oil and Vinegar), and STS (Stepwise Triangular System). Our new trapdoor has properties which are in between MIA on the one hand and STS on the other hand.

It has practical value in that we can expect it to run faster, especially in resource-limited environments (e.g. smart cards). Due to its structure, we call it "ℓ-Invertible Cycles" (ℓIC). Before motivating this name, we will first introduce the trapdoor.

2.1 Basic Trapdoor

A *Cremona Transformation* is a map on the projective plane that is quadratic in the homogeneous coordinates [Ful89]. A standard example is the map $(A_1, A_2, A_3) \to (A_2A_3, A_3A_2, A_1A_2)$ which easily checks to be well-defined. The map is uniquely and efficiently invertible when $A_1A_2A_3 \neq 0$.

We extend this idea below to any integral cycle length $\ell \geq 2$; we illustrate with the case $\ell = 3$ since (unfortunately) the case $\ell = 2$ is a bit more technical.

Note that we write $\mathbb{N}$ for the non-negative integers, *i.e.*, we have $\mathbb{N} := \mathbb{Z}^+ \cup \{0\}$. To express properly the successor in $\{1, \ldots, \ell\}$ we define

$$\mu : \{1, \ldots, \ell\} \to \{1, \ldots, \ell\} \quad : \quad \mu(i) := \begin{cases} 1 & \text{for } i = \ell \\ i+1 & \textit{otherwise} \end{cases} \tag{2}$$

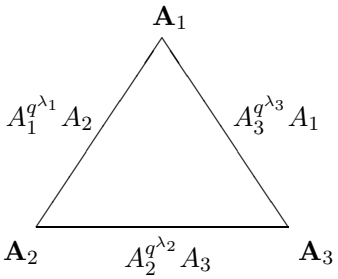


Figure 2: Graphical Representation of 3-Invertible Cycles

DEFINITION 2.1 *Fix an integer $\ell \geq 2$ as the length of the cycle. Let $\mathbb{F}$ be the base field with $q := |\mathbb{F}|$ elements and $\mathbb{E} :=GF(q^k)$ its k^{th}-degree extension for some $k \in \mathbb{Z}^+$. Computations in $\mathbb{E}$ are modulo the irreducible polynomial $\pi(t) \in \mathbb{F}[t]$. We denote $Q := |\mathbb{E}| = q^k$ and have $m = n = \ell k$ for the number of variables and equations over the ground field $\mathbb{F}$, respectively. In addition, let $S, T \in Aff^{-1}(\mathbb{F}^n)$ be two invertible affine mappings and the vector $\Lambda := (\lambda_1, \ldots, \lambda_\ell) \in \{0, \ldots, k-1\}^\ell$. We now have the following mapping:*

$$P : \mathbb{E}^\ell \to \mathbb{E}^\ell : (A_1, \ldots, A_\ell) \to (A_1^{q^{\lambda_1}} A_2, \ldots, A_{\ell-1}^{q^{\lambda_{\ell-1}}} A_\ell, A_\ell^{q^{\lambda_\ell}} A_1) \tag{3}$$

Identifying the corresponding coefficients in the vector spaces $\mathbb{F}^n$ and $\mathbb{E}^\ell$, we get a canonical bijection

$$\phi : \mathbb{F}^n \to \mathbb{E}^\ell \quad : \quad (x_1, \ldots, x_n) \to (x'_1 + x'_2 t + \ldots x'_k t^{k-1}, \ldots, x'_{n-k+1} + x'_{n-k+2} t + x'_n t^{k-1}) \tag{4}$$

and its inverse ϕ^{-1}. The public key is computed as the composition

$$\mathcal{P} : \mathbb{F}^n \to \mathbb{F}^m \quad : \quad \mathcal{P} := T \circ \phi^{-1} \circ P \circ \phi \circ S\,. \tag{5}$$

We then call such a $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key system of the ℓIC-type.

The name "invertible cycle" is motivated by the structure of P as the variables $A_1, \ldots, A_\ell$ can be drawn in the form of a cycle, see Figure 2 for the case $\ell = 3$. The variables $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3$ are the nodes while each edge stands for a product $A_i^{q^{\lambda_i}} A_{\mu(i)}$ with $i = 1, 2, 3$.

Note that the use of the canonical bijection ϕ is similar for the Matsumoto-Imai Scheme A (MIA) and Hidden Field Equations (HFE). However, we have $\ell = 1$ here, and also a different form of the central mapping $P \in \mathbb{E}[X]$. In the sequel, we denote the output of P by

$$B_1 := A_1^{q^{\lambda_1}} A_2, \ldots, B_{\ell-1} := A_{\ell-1}^{q^{\lambda_{\ell-1}}} A_\ell, B_\ell := A_\ell^{q^{\lambda_\ell}} A_1$$

REMARK 2.2 *The mapping $A_i^{q^{\lambda_i}}$ is linear over the ground field $\mathbb{F}$. Hence, the central equation P can be expressed as a system of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomials over the ground field $\mathbb{F}$.*

REMARK 2.3 *Replacing $A_i^{q_i^\lambda} A_{\mu i}$ by $A_i^{q^{\lambda_i}} A_{\mu(i)}^{q^{\kappa_i}}$ for $1 \leq i \leq \ell$ and some $\kappa_i \in \mathbb{N}$ does not increase the security of ℓIC: we can always reduce the second expression to $A_i^{q^{\lambda_i - \kappa_i \pmod k}} A_{\mu(i)}$ by using Frobenius transformations. In a nutshell, we exploit that Frobenius transformations are invertible linear mappings over the vector spaces $\mathbb{F}^n$ and $\mathbb{F}^k$, respectively, and can hence be "absorbed" into the mappings $S, T \in Aff^{-1}(\mathbb{F}^n)$. For $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic system, this idea has been introduced under the name* Frobenius sustainers *[WP05a].*

2.2 Singularities

To use ℓIC in as an encryption or as a signature scheme, we need to invert the central map P, *i.e.*, we need to find a solution $(A_1, \ldots, A_\ell) \in \mathbb{E}^\ell$ for given input $(B_1, \ldots, B_\ell) \in \mathbb{E}^\ell$. Unfortunately, this is not possible in all cases; due to its form ℓIC has following singularities:

$$\{ (A_1, \ldots, A_\ell) \in \mathbb{E}^\ell \mid A_1 = 0 \vee \ldots \vee A_\ell = 0\}$$

Having $Q := |\mathbb{E}|$ and exploiting that Q in contrast to ℓ is usually "big" for practical and secure schemes we can approximate the probability that a singularity occurs by

$$\left(\sum_{i=1}^{\ell} (Q-1)^{\ell-1} \right) / Q^\ell \approx \frac{\ell}{Q}$$

In the Matsumoto-Imai Scheme A, we do not have this problem as MIA forms a bijection. In comparison, Hidden Field Equations does not allow to compute an inverse in about 40% of all cases for a practical choice of parameters [Pat96, CGP01, WP04]. Our new trapdoor ℓIC is hence between these two extreme cases. Practical values for Q will be discussed in Sec. 5.

2.3 Inversion

As we have as many free variables A_i as conditions B_i for $1 \leq i \leq \ell$, we may expect one solution on average when inverting P. Alas, this is not always true, as shown by the obvious counterexample:

$$(B_1, B_2) := P(A_1, A_2) := (A_1 A_2, A_2 A_1) \in \mathbb{E}^2.$$

So there are instances of ℓIC that cannot be inverted usefully. For practical use, we construct below a sequence of specific ℓIC instances which allows easy inversion.

Lemma 2.4 *For a fixed $\ell \geq 2$, let our ℓIC central map $P : (A_1, \ldots, A_\ell) \mapsto (B_1, \ldots, B_\ell)$ be*

$$B_1 := \begin{cases} A_1 A_2 & \text{for } \ell \text{ odd and} \\ A_1^q A_2 & \text{for } \ell \text{ even} \end{cases},$$

$$B_i := A_i A_{\mu(i)} \text{ for } 2 \leq i \leq \ell\,.$$

Then the inverse image of $(B_1 \ldots B_\ell)$, where $B_i \in E^* := E \backslash \{0\}$ for $i = 1 \cdots \ell$, is given by

$$A_1 := \begin{cases} \sqrt{\frac{\prod_{i=0}^{(\ell-1)/2} B_{2i+1}}{\prod_{i=1}^{(\ell-1)/2} B_{2i}}} & \text{for } \ell \text{ odd and} \\ \sqrt[q-1]{\frac{\prod_{i=0}^{\ell/2-1} B_{2i+1}}{\prod_{i=1}^{\ell/2} B_{2i}}} & \text{for } \ell \text{ even} \end{cases}$$

$$A_i := \frac{B_i}{A_{\mu(i)}} \text{ for } i = \ell, \ldots, 2$$

Proof.

Case $\ell = 3$: We have $B_1 := A_1 A_2, B_2 := A_2 A_3, B_3 := A_3 A_1$. Simple computations yield the required result $A_1 := \sqrt{B_1 B_3 / B_2}, A_3 := B_3 / A_1, A_2 := B_2 / A_3$.

Case ℓ odd, $\ell > 3$: We use induction to extend the result from $\ell = 3$ to all odd $\ell > 3$. Therefore we observe that the structure of the central mapping P allows us to write equations of the form $A_i = A_{\mu(\mu(i))}\frac{B_i}{B_{i+1}}$ for $1 < i < \ell$ by eliminating the variable $A_{\mu(i)}$. Hence, the fraction $\frac{B_i}{B_{i+1}}$ can be inserted in the inversion formula for A_1 in the case $(\ell - 2)$.

Case ℓ even: The proof for this case is analogous. We start our induction with $\ell = 2$ and have $B_1 := A_1^q A_2, B_2 := A_2 A_1$ and its inverse $A_1 := \sqrt[q-1]{B_1/B_2}, A_2 := B_2/A_1$.

□

Bijectivity. For ℓ odd and $\mathbb{F}$ of characteristic 2, the above mapping is a bijection in $(\mathbb{E}^*)^\ell$. For ℓ even, the situation is more difficult as $(q-1) \mid (q^a - 1)$ for any $a \in \mathbb{Z}^+$, and we loose bijectivity for any $q > 2$. However, for $q = 2$, we obtain a bijection. Moreover, inversion now only costs two divisions in the extension field $\mathbb{E}$ and we need not solve any nontrivial equations.

Special instances. We give specific names to some ℓIC instances given by the formulæ in Lemma 2.4. These will come in useful when constructing practical schemes. We call the case $\ell = 2$ *Binary Invertible Cycle* (BIC). The case $\ell = 3$ is called *Delta Invertible Cycle* (DIC) due to the form of the corresponding cycle, see Fig. 2. We may call the $\ell = 4$ and $\ell = 5$ cases analogously the Square and Penta Invertible Cycles (or SIC and PIC). We will now obtain some cryptanalytic properties of ℓIC before constructing actual schemes and discussing an optimal choice for ℓ.

Extensions. Extensions of this idea are given in App. A.

3 Cryptanalytic Properties of ℓIC

We herein discuss some basic cryptanalytic properties of the new trapdoor. This serves a dual purpose: We find an easy cryptanalysis for ℓIC in its basic form. Simultaneously, we effectively put ℓIC through the same screening process as other $\mathcal{MQ}$ trapdoors, particularly Matsumoto Imai Scheme A. This points us toward ways to build practical, more resilient ℓIC-based schemes.

3.1 Patarin Relations

We start with an extension of the Patarin relations used to cryptanalyse MIA [Pat95]. This was used by Fouque, Granboulan, and Stern to cryptanalyse the internally perturbed MIA encryption scheme (PMI/MIAi) [FGS05]. As is more customarily employed against symmetric cryptosystems, we examine this multivariate differential :

$$\begin{aligned} P(A_1, \ldots, A_\ell) - P(A_1 - \delta_1, \ldots, A_\ell - \delta_\ell) + P(\delta_1, \ldots, \delta_\ell) \\ = (A_1^{q^{\lambda_1}}\delta_2 + A_2\delta_1^{q^{\lambda_1}}, \ldots, A_\ell^{q^{\lambda_\ell}}\delta_1 + A_1\delta_\ell^{q^{\lambda_\ell}}) \end{aligned}$$

We observe that the above equations are linear in the unknowns $A_i \in \mathbb{E}$ for any given values $\delta_i \in \mathbb{E}$ and $1 \le i \le \ell$. Now we simply pick δ_i at random and compute many differentials of the public key. Soon we recover enough linear relations to invert the public map. This effectively finds an equivalent private key. App. C estimates the number of linearization equations for $\mathbb{F} = \mathrm{GF}(2)$.

This resembles MIA and HFE in that the Patarin attack is very efficient against the former, and an extended version of the attack defeats the latter if bijective central maps are used [Pat95, Pat96].

3.2 Rank Attacks

In a *rank attack*, the quadratic parts central and public polynomials of a given $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key system are written as symmetric matrices. We try to recover the private key by finding linear combinations of the public matrices with certain specific ranks. They are first introduced to cryptology by Coppersmith, Stern, and Vaudenay in the cryptanalysis of Birational Permutations [CSV93]. See the later [GC00, WBP04, YC05] for further extensions and analysis.

Very elegant (in terms of pure algebra) versions have been presented. However Goubin and Courtois in [GC00] has the most straightforward exposition of rank attacks. There are two distinct types: In one the cryptanalyst randomly tries to hit kernel vectors of a linear combination of the public matrices with the lowest rank R. The running time is proportional to $q^{R\lfloor m/n \rfloor}$. In the other random linear combination are taken, hoping to locate a precipitous fall in rank. This takes time proportional to q^u, where u counts the central equations whose coefficients must vanish.

For ℓIC, we want to write matrices in blocks corresponding to pairs of variable in the larger field $\mathbb{E}$. Express central matrices as $H_1, \ldots, H_\ell \in \mathbb{E}^{\ell \times \ell}$ and their $\mathbb{E}$-blocks as $\eta_{i,j,k} \in \mathbb{E}$ for $1 \leq i, j, k \leq \ell$.

$$\eta_{i,j,k} := \begin{cases} M_{\lambda_i} & \text{if } i = j, k = \mu(i) \\ M_{\lambda_i}^T & \text{if } i = k, j = \mu(i) \\ 0 & \textit{otherwise} \end{cases}$$

Where M_r is the matrix in $\mathbb{F}^{k \times k}$ that correspond to the Frobenius map $A \mapsto A^{q^r}$. Note that these matrices are symmetric. In the case of DIC, *i.e.*, $\ell = 3$, they effectively specialize to

$$H_1 := \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \; H_2 := \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \; H_3 := \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

All these matrices have essentially rank 2 over the extension field $\mathbb{E}$. For the actual attack, we would need to transfer $M \in \mathbb{E}^{\ell \times \ell}$ to $\mathbb{F}^{n \times n}$. However the overall attack complexity is not affected by this change of vector space. Just as in other schemes using extension fields (e.g. cf. MFE [WYHL06]), when performed in $\mathbb{F}$ we have a rank of $2k$ for all these matrices. We may see that the running time of the both the above algorithms are $Q^2 = q^{2k}$ times some polynomial factor in n and m. This factor is the order of $O(n^\omega)$ where ω is the order of matrix multiplication. In practice, the size of the matrices are small enough that we can take $\omega = 3$ (Gauss).

Note that there are instances in which one or the other rank attack simply fails to work. One example is the case of BIC, *i.e.*, for $\ell = 2$. Here rank attacks will not apply as any nontrivial linear combination of the private polynomials (matrices) have the maximum rank $n = 2k$. We have checked rank attacks against ℓIC with some small scale examples.

3.3 Gröbner Basis Computations

Another important attack are Gröbner attacks as used against Hidden Field equations by Faugère and Joux [FJ03]. The algorithms of the Faugère-Lazard type essentially involves doing eliminations on an extended Macaulay matrix. These includes the methods $\mathbf{F_4}/\mathbf{F_5}$ and the related algorithm that is known in crypto circles as XL [CKPS00, Fau99, Fau02] plus their variations.

We know from the cryptanalysis of MIA and HFE that their easy algebraic structure leads to a low running time of the corresponding Gröbner algorithm. Due to the very easy structure of ℓIC, we expect a similar behaviour here. This is in line small scale experiments using Magma [MAG]. A modifier is needed to disrupt the regular structure. In general, when the structure of the system is sufficiently perturbed, the behavior is as studied by Bardet, Faugère *et al* [BFS04, BFSY05, YC04b, YC04a]. Some further details are given in the Appendix B.

3.4 Separation of Oil and Vinegar

In the original DIC, we see that variables corresponding to the components of A_1 is only multiplied with those of A_2 and A_3. this makes for a UOV type of attack [KPG99] which has a complexity roughly proportional to n^4q^d, where d is the difference between the size of the oil and vinegar sets. We can proceed similarly for other choices of ℓ. We see that the UOV attack has time complexity $\sim Q$ for odd ℓ and very small complexity for even ℓ.

This would be enough to do any ℓIC encryption scheme in except that the "plus" modifier disrupt the structure so that the UOV attack does not work; the minus modifier does not change the complexity of the UOV attack, so for the parameters we choose below UOV is conjectured not to be a problem. This is consistent with small-scale tests.

3.5 Branching

The original MIA/C* scheme of Matsumoto and Imai used a technique called "branching" to obtain higher speed. At first glance, branching looks quite similar to the idea of ℓIC: operations in a big extension field $\mathrm{GF}(q^n)$ are replaced by operations in smaller extension fields $\mathrm{GF}(q^k)$ for some $k \ll n$. The most efficient algorithm to break branching we are aware of is from Felke [Fel04]. It separates the different branches of a given system in $O(n^6)$. However, there is an important difference between ℓIC and branching: the operations in the different branches of C* were independent while there is a strong interaction between the different components of ℓIC. This statement remains true even for the restricted version of Lemma 2.4. We have investigated this matter and concluded that the attacks against branching do not apply against ℓIC. Moreover, there is no known way to attack modified versions of MIA — either with the minus modification (cf Sec. 4.1) or with internal perturbation (cf Sec. 4.3).

3.6 Further Attacks

Another kind of attacks are algorithms of the XL family to solve systems of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations over finite fields [CKPS00]. Due to recent work [AFI+04, YC04a] we know these algorithms are variants of known Gröbner basis algorithms. A different class are algorithms from [CGMT02] which deal with the case $n \gg m$. As we usually have $m = n$, or $n \approx m$ for the embedding modification (cf Sec. 4.4), these algorithms are not applicable to our setting.

Nevertheless, due to the effectiveness of the attacks considered above, we need to apply modifiers [WP05b, Sec. 4] to the basic trapdoor to obtain secure schemes. This is the same situation as for MIA and HFE. There may be other attacks, but to the best of our knowledge we have named every known attack against a system of this type.

4 Modified Versions

4.1 ℓ-Invertible Cycles Minus (ℓIC-)

The first modification is the so-called "minus" modification. Here, we use a *reduction* or *projection*

$$R : \mathbb{F}^{r+m} \to \mathbb{F}^m : (y_1, \ldots, y_m) := (y_1, \ldots, y_{m+r})$$

and have $r := n - m$ as "reduction parameter". The public key is now constructed as

$$\mathcal{P} := R \circ T \circ \phi^{-1} \circ P \circ \phi \circ S\,.$$

In contrast to (5), we have inserted the reduction R after the affine transformation T. In effect, this means that we drop the last r equations. When inverting ℓIC, we assign random values to these missing r coordinates over $\mathbb{F}$. Hence, we have q^r possible inputs for each message $y \in \mathbb{F}^m$.

As for Matsumoto-Imai Scheme A and Hidden Field Equations, the effect of the minus modification is two-fold. First, it increases the complexity of the Patarin attack (Sec. 3.1) by a factor of q^r. The argument is the same as for the original Patarin-attack against MIA: instead of one possible solution, the attacker now faced with an r-dimensional vector space over $\mathbb{F}$ of possible solutions. To our current knowledge, picking the right one requires brute force and hence at least q^r operations. Second, the attack complexity of the Faugère-Joux attack [FJ03] also increases by at least q^r. The matrix algebra adds another factor of n^ω, cf. Sec. 3.2.

In effect, the situation resembles that of MIA/C^*. We cannot use ℓIC- for encryption schemes but only as signature schemes: as there are r equations missing, the legitimate user has the same workload for recovering the correct solution $x \in \mathbb{F}^n$. As our security assumption is that q^r computations are not possible, we reached a contradiction if we assume that the legitimate user can obtain the message x while the attacker cannot.

As for Stepwise-Triangular Systems, the rank attack is not affected by the minus modification. Hence, every construction using ℓIC- has to make sure that this attack is infeasible.

4.2 ℓ-Invertible Cycles Plus (ℓIC+)

The generic plus modification adds $a \in \mathbb{Z}^+$ random equations in n input variables each to the private key. As there is no trapdoor for these equations, they will slow down signature generation by q^a as we only have a chance of q^{-a} that these additional conditions on the output of the ℓIC mapping P will be met. For encryption, we do not have such a problem as these equations are fulfilled by definition.

More formally, we consider $\mathcal{P}^* := \phi^{-1} \circ P \circ \phi$ the ℓIC mapping over the ground field $\mathbb{F}$ and $\hat{\mathcal{P}} \in_R \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^a)$ a system of a polynomials in n variables each, uniformly randomly chosen from the set $\mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^a)$. Moreover, we have $T \in \mathrm{Aff}^{-1}(\mathbb{F}^{n+a})$ and consequently $m := n + a$ for the number of equations. Defining $(p'_1, \ldots, p'_n) := \mathcal{P}^*$ and $(p'_{n+1}, \ldots, p'_{n+a}) := \hat{\mathcal{P}}$ for the individual polynomials of the ℓIC mapping $\mathcal{P}^*$ and the a random polynomials from $\hat{\mathcal{P}}$, we can express the new trapdoor as

$$\mathcal{P} : \mathbb{F}^n \to \mathbb{F}^m : \mathcal{P} := T \circ \mathcal{P}' \circ S$$

for $\mathcal{P}' := (p'_1, \ldots, p'_n, p'_{n+1}, \ldots, p'_{n+a})$.

Patarin relations and Gröbner attacks are not affected by the plus modification. However, it is still useful to build an encryption scheme. In the case of MIA because the "plus" helps to overcome some attacks against the internally perturbated modification. Here, it also prevents a UOV attack.

4.3 ℓ-Invertible Rings Internally Perturbated (ℓICi)

Internal perturbation has been introduced for MIA under the name Perturbated Matsumoto-Imai — PMI [Din04]. It is also known under the name MIAi. Moreover, internal perturbation has been used with HFE (ipHFE or HFEi) [DS05a]. As PMI/MIAi has been broken in [FGS05], a new variant PMI+/MIAi+ has been proposed [DG05]. Due to space limitations in this paper we do not go into details, but we want to stress that PMI+ is not affected by the attack from [FGS05]. Hence, combining the two modifications *internal perturbation* and *plus* allows the construction of an efficient encryption scheme. However, there is one condition: the central mapping $\mathcal{P}'$ and all

its components need to have full rank. In our setting, this means that we we cannot use any other cycle length but $\ell = 2$, *i.e.*, BIC.

After talking about the impact of the internal perturbation modification, we now properly introduce it: Let $w \in \mathbb{Z}^+$ for $w < n$ be the perturbation dimension, $\mathcal{P}^i \in_R \mathcal{MQ}(\mathbb{F}^w, \mathbb{F}^n)$ a uniformly randomly chosen system in w input variables and n equations, and $S^i \in \mathrm{Aff}^{-1}(\mathbb{F}^n, \mathbb{F}^w)$ the so-called "perturbation space". Note that the perturbation space has the same input variables $x_1, \ldots, x_n$ as the affine transformation $S \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$. However, it has only dimension w. Hence we can write $(z'_1, \ldots, z'_w) := S^i(x_1, \ldots, x_n)$ for the perturbation variables $z'_1, \ldots, z'_w$. As for the plus modification, we denote with $\mathcal{P}^* := \phi^{-1} \circ P \circ \phi$ the ℓIC mapping over the ground field $\mathbb{F}$.

The public key for ℓICi is now composed as

$$\mathcal{P} := T \circ [(\mathcal{P}^* \circ S) + (\mathcal{P}^i \circ S^i)] ,$$

i.e., we add the perturbation polynomials to the original ℓIC-polynomials. To invert this modified trapdoor, *i.e.*, to compute $x \in \mathbb{F}^n$ for given $y \in \mathbb{F}^m$, we need to guess correctly the values of the perturbation variables $(z'_1, \ldots, z'_w) \in \mathbb{F}^w$ — which translates to a workload proportional to q^w. As the number of equations and the number of variables matches, we expect one solution on average for any given input $y \in \mathbb{F}^m$. However, when used as an encryption scheme, there is at least one valid output $x \in \mathbb{F}^n$. We know that the i modifier by itself is not secure, and it must be combined with the + modifier as shown by the F-G-S differential attack [FGS05].

4.4 ℓ-Invertible Cycles Embedded (ℓIC↗) without Singularities

In this section, we introduce the previously not known modifier embedding (↗). It is motivated by the practical need to avoid singularities in trapdoors of the ℓIC-type.

When combined with the minus modification, singularities are of no concern: they are too few and we can always change the input in the missing equations to obtain a possible signature. However, when ℓIC is used in the context of an encryption scheme, its singularities pose a problem as they lead to decryption failures. The modification described in this section can also be used in other schemes which suffer from a decryption failure such as [WYHL06]. In fact, it is a new generic modifier and can be used in any $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic construction.

For our new embedding modifier we embedding the following translation:

$$\begin{array}{rcl} \mathbb{F}^{k-1} & \to & \mathbb{F}^k \\ (x_1, \ldots, x_{k-1}) & \to & (x_1, \ldots, x_{k-1}, 1) \end{array}$$

In effect, we have eliminated the zero-point from the vector space $\mathbb{F}^k$. As we used the canonical bijection ϕ between the vector space $\mathbb{F}^k$ and the extension field $\mathbb{E}$, the zero of $\mathbb{E}$ cannot be reached anymore for any given input $(x_1, \ldots, x_{k-1}) \in \mathbb{F}$. The price we pay are less input variables, *i.e.*, we now obtain an overdetermined system of polynomials.

The above idea can be easily extended to all ℓ variables $A_1, \ldots, A_\ell \in \mathbb{E}$. Calling the corresponding transformation $\nu : \mathbb{F}^n \to \mathbb{F}^{n-\ell}$ and setting $k := (n - \ell)/\ell$ for $k \in \mathbb{N}$ we obtain the following construction for the public key

$$\mathcal{P} = T \circ \phi^{-1} \circ P \circ \nu \circ S . \tag{6}$$

To obtain a singularity free ℓIC for signing, the "inverse" transformation

$$\nu^{-1} : (y_1, \ldots, y_{k-1}, 1) \to (y_1, \ldots, y_{k-1})$$

needs to be inserted between the affine transformation T and the ℓIC mapping P. To the same effect, we could have used the construction of (6). However, this would have slowed down signature generation by a factor of q^ℓ as we have ℓ additional equations over $\mathbb{F}$ to satisfy for any given input $B_1, \ldots, B_\ell \in \mathbb{E}$.

5 Practical Instances

Herein we use the cryptanalytic results from the previous section to develop practical instances of ℓIC. Main purpose is to see how variations on ℓIC scales up for different security levels.

5.1 Signature

To obtain a secure signature scheme, we use ℓIC- as this seems the most suitable modification for our purpose. In particular, the security of the minus modification is well understood; we are therefore able to give instances of ℓIC for several security levels. Different choices of parameters are discussed in App. D. We restrict to the case $q = 256$ as this allows efficient implementation on 8-bit microprocessors which are still dominant in low-end smart cards. We summarize optimal choices in Table 1. It is very different for encryption, which everyone believes to be more difficult.

Table 1: ℓIC- over GF(256) with Different Security Levels for Signing

Claimed	Input	Output			Parameters			Attack Complexity		Key Size [kBytes]	
Security	[bits]	[bits]	n	m	ℓ	k	r	Gröbner	Rank/UOV	Public	Private
2^{80}	160	240	30	20	3	10	20	2^{80}	2^{85}	9.92	1.86
2^{96}	192	288	36	24	3	12	24	2^{96}	2^{104}	16.8	2.59
2^{128}	256	384	48	32	3	16	32	2^{130}	2^{137}	39.20	4.70

5.2 Encryption

We base our proposed encryption scheme on BICi+↗, *i.e.*, 2-Invertible Cycles with internal perturbation, added equations, and embedding. We have the following reasons for this choice: first, we cannot use ℓIC or BIC in its original form, due to Patarin attacks. Second, to use internal perturbation, we need ℓIC with full rank. This implies $\ell = 2$ and hence BIC. Moreover, internal perturbation alone is not secure, due to the efficient attack from [FGS05]. Therefore, we need the plus modification. Last but not least we want to avoid decryption errors and hence include the embedding modification. With this choice of scheme, we suggest the following parameters: $q = 2, n = 132, m = 146, \ell = 2, k = 67, w = 6, a = 12$. This leads to a public key of 160.2 kBytes and a private key of 5.7 kBytes, respectively. The claimed security level is 2^{80}. Our choice of parameters is based on [DG05]. Due to space limitations in this paper we do not repeat their arguments but point to [DG05]. However, we want to stress that at present, our understanding of the security of the internal perturbation modification is limited although there some results on Gröbner bases in [DGS$^+$05]. This means in particular that we do not have precise security estimations for higher security levels.

5.3 Implementation and Speed

A good overview on implementing finite field operations can be found in [LD00]. Computing direct division in finite fields is given in [FW02]. Counting operations for the inversion formula in Lemma 2.4 over $\mathbb{E} = \mathrm{GF}(q^k)$, we see that we need ℓ division, $(\ell - 2)$ multiplications, and one root. Note that the operations do not take place in a big field $\mathrm{GF}(q^n)$ but in a much smaller extension field $\mathrm{GF}(q^k)$. It is difficult to give a closed formula for the speed of basic arithmetic operations as they largely depend on the model used, *e.g.*, hardware vs. software, operations on bits vs. operations on processor words. Nevertheless, when counting our costs in operations in the ground field $\mathbb{F}$, we can roughly say that we have $O(a^2)$ for squaring/multiplying and $O(a^3)$ for division/exponentiation. Here we have $l \in \mathbb{Z}^+$ the extension degree of the corresponding field $\mathbb{E} = \mathrm{GF}(q^a)$ over the ground field $\mathbb{F} = \mathrm{GF}(q)$. We have to keep this in mind when comparing ℓIC with the other two mixed field schemes MIA and HFE.

Comparison with MIA and HFE. Inverting the mixed field scheme MIA costs one exponentiation with large exponent [CGP02]. In a nutshell, this translates to n squaring operations and $1/2n$ multiplications in $\mathrm{GF}(q^n)$. Therefore, we obtain an overall workload of $O(n^3)$. Tricks to speed this operation up can be found in [ACDG03]. In the case of HFE, the situation is even worse as we need to execute a complete root finding algorithm to invert the central mapping [CGP01]. Its running time is estimated to be in $O(n^3d^2 + n^2d^3)$ for d the total degree of the central mapping [Pat96]. In practice, we have $d = 129 \ldots 257$.

We can summarize our results for the three maps MIA, HFE, and ℓIC as follows: the first needs $O(n^3)$ operations in the ground field $\mathbb{F}$ for n the extension degree as it needs to compute Y^h for given $Y \in \mathrm{GF}(q^n)$ and $h \in \mathbb{Z}^+$, *i.e.*, an exponentiation. The second needs to solve a univariate polynomial equation $P(X) = Y$ for P being a polynomial of fixed degree $d \in \mathbb{Z}^+$. The corresponding running time is about $O(n^3d^2 + n^2d^3)$ operations in the ground field $\mathbb{F}$. Finally, ℓIC needs $O(\ell k^3 + \ell k^2 + d^2k^3 + d^3k^2)$ operations over the ground field $\mathbb{F}$ although some choices of ℓ, d allow a lower running time (see above).

A choice for MIA is Sflash$^{\mathrm{v2}}$ with $q = 128$, $n = 37$ [CGP02]. For HFE, we have $q = 2$, $n = 103$ in Quartz [CGP01]. Choices for ℓIC are given in Sec. 5.2 and Table 1, respectively. Both trapdoors have a claimed security level of 2^{80} 3DES computations as required in NESSIE [NES]. Note that Quartz uses the underlying trapdoor four times to achieve very short signatures of 128 bit. This special construction is called a "Chained Patarin Construction" (CPC). We summarize our comparison in Table 2. preliminary runs shows that signing with $m = 24, n = 36$ is at least 3–5 times as fast as SFLASH (which also means it is faster than enTTS [YC05]).

Further Speed up. Using two arithmetic units, we can speed up the above inversion: as soon as we have the value for A_1, we can compute the values A_i for $i \geq 2$ using the cycle in the other direction.

6 Conclusions

In this article, we have constructed a new basic $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic trapdoor called ℓ-invertible cycles (ℓIC). It is the first time since nearly a decade that a basic trapdoor has been found. The main motivation for this new trapdoor is speed: instead of computing operations in the big finite field $\mathbb{E} = \mathrm{GF}(q^n)$ for $q := |\mathbb{F}|$ and n the number of variables, we compute in the much smaller extension field $\mathbb{E} = \mathrm{GF}(q^k)$ for $n = \ell k$ for some cycle length ℓ. Typical choices of ℓ are $2 \ldots 6$. Depending on the architecture, finite field arithmetic costs up to $O(n^3)$. Hence, decreasing

the size of the extension field $\mathbb{E}$ results in a significant speed-up in practice. In particular, our implementation is expected to outperform the previously fasted trapdoor Matsumoto-Imai Scheme A (MIA). In addition, we have formally introduced the new embedding modifier ($\nearrow$). It is motivated by the practical need to achieve ℓIC-type schemes without decryption failure. Apart from ℓIC, constructions like [WYHL06] suffer from this problem.

Table 2: Mixed Field Trapdoors with Claimed Security Level 2^{80}

Trapdoor	Complexity to Invert Trapdoor	Parameters	Key Size [kBytes] Public	Private
HFE (Quartz)	$O(n^3d^2 + n^2d^3)$	$q = 2$, $n = 103$, $d = 129$	71	3
MIA (Sflash)	$O(n^3)$	$q = 128$ $n = 37$	15.4	2.45
ℓIC, $\ell = 3$	$O(\ell k^3 + \ell k^2 + d^2k^3 + d^3k^2)$	$q = 256$, $k = 10$, $d = 255$	9.92	1.86

Table 2 shows the different complexities, parameters and public key sizes for trapdoors of the mixed field types with a claimed security level of 2^{80}. Unfortunately, we do not have exact estimations on their inherent complexity but asymptotic ones. Nevertheless, we see that ℓIC for a similar security level is expected to perform significantly better than the two other basic trapdoors HFE (using parameters from Quartz) and MIA (parameters from Sflash^{v2}). Apart from this, we have shown that ℓIC can be used both in signature schemes of various security levels as well as in an encryption scheme. We want to stress here that trapdoors from the single field class, *i.e.*, Unbalanced Oil and Vinegar (UOV) and Stepwise-Triangular Schemes (STS) do *not* allow constructions leading to encryption schemes.

So as an overall conclusion, we have presented a new trapdoor which is both interesting from a theoretical point of view and also has advantages over previously known schemes. At present we have to leave it as an open question if other forms of ℓIC than these given in Lemma 2.4 allow efficient inversion. Some leads to this question have been given in App. A.

We stress that it is still an original sin that no list of possible attacks can be exhaustive. $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic schemes are still in need of some provable security results. But we hope to have shown that the variety available in the genre keeps it in play and interesting.

References

[ACDG03] Mehdi-Laurent Akkar, Nicolas T. Courtois, Romain Duteuil, and Louis Goubin. A fast and secure implementation of SFlash. In *Public Key Cryptography — PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 267–278. Y. Desmedt, editor, Springer, 2002.

[AFI+04] Gwénolé Ars, Jean-Charles Faugère, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita. Comparison between xl and gröbner basis algorithms. In *Advances in Cryptology — ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 338–353. Pil Joong Lee, editor, Springer, 2004.

[BFS04] M. Bardet, J.-C. Faugère, and B. Salvy. On the complexity of gröbner basis computation of semi-regular overdetermined algebraic equations. In *Proceedings of the International Conference on Polynomial System Solving*, pages 71–74, 2004.

[BFSY05] M. Bardet, J.-C. Faugère, B. Salvy, and B.-Y. Yang. Asymptotic expansion of the degree of regularity for semi-regular systems of equations. In P. Gianni, editor, *MEGA 2005 Sardinia (Italy)*, 2005.

[CGMT02] Nicolas Courtois, Louis Goubin, Willi Meier, and Jean-Daniel Tacier. Solving underdefined systems of multivariate quadratic equations. In *Public Key Cryptography — PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 211–227. David Naccache and Pascal Paillier, editors, Springer, 2002.

[CGP01] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Quartz: Primitive specification (second revised version)*, October 2001. `https://www.cosic.esat.kuleuven.be/nessie` Submissions, Quartz, 18 pages.

[CGP02] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Sflash: Primitive specification (second revised version)*, 2002. `https://www.cosic.esat.kuleuven.be/nessie`, Submissions, Sflash, 11 pages.

[CKPS00] Nicolas T. Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Bart Preneel, editor, Springer, 2000. Extended Version: `http://www.minrank.org/xlfull.pdf`.

[Cr93] Douglas R. Stinson, editor. *Advances in Cryptology — CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*. Springer, 1993. ISBN 3-540-57766-1.

[CSV93] Don Coppersmith, Jacques Stern, and Serge Vaudenay. Attacks on the birational permutation signature schemes. In Cr [Cr93], pages 435–443.

[DG05] Jintai Ding and Jason E. Gower. Inoculating multivariate schemes against differential attacks. In *Public Key Cryptography — PKC 2006*, Lecture Notes in Computer Science. Springer, 2005. 12 pages, *to appear at PKC'06*, preliminary version at `http://eprint.iacr.org/2005/255`.

[DGS+05] Jintai Ding, Jason E. Gower, Dieter Schmidt, Christopher Wolf, and Z. Yin. Complexity estimates for the F_4 attack on the perturbed Matsumoto-Imai cryptosystem. In *Cryptography and Coding - 10th IMA International Conference*, volume 3796 of *Lecture Notes in Computer Science*, pages 262–277. Nigel P. Smart, editor, Springer, 2005.

[Din04] Jintai Ding. A new variant of the Matsumoto-Imai cryptosystem through perturbation. In *Public Key Cryptography — PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 305–318. Feng Bao, Robert H. Deng, and Jianying Zhou (editors), Springer, 2004.

[DS05a] Jintai Ding and Dieter Schmidt. Cryptanalysis of HFEv and internal perturbation of HFE. In PKC [PKC05], pages 288–301.

[DS05b] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *Conference on Applied Cryptography and Network Security — ACNS 2005*, volume 3531 of *Lecture Notes in Computer Science*, pages 164–175. Springer, 2005.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra*, 139:61–88, June 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *International Symposium on Symbolic and Algebraic Computation — ISSAC 2002*, pages 75–83. ACM Press, July 2002.

[FD85] Harriet Fell and Whitfield Diffie. Analysis of public key approach based on polynomial substitution. In *Advances in Cryptology — CRYPTO 1985*, volume 218 of *Lecture Notes in Computer Science*, pages 340–349. Hugh C. Williams, editor, Springer, 1985.

[Fel04] Patrick Felke. On the affine transformations of HFE-cryptosystems and systems with branches. Cryptology ePrint Archive `http://eprint.iacr.org`, Report 2004/367, 2004. `http://eprint.iacr.org/2004/367`, version from 2004-12-17, 10 pages.

[FGS05] Pierre-Alain Fouque, Louis Granboulan, and Jacques Stern. Differential cryptanalysis for multivariate schemes. In *Advances in Cryptology — EUROCRYPT 2005*, Lecture Notes in Computer Science. Ronald Cramer, editor, Springer, 2005. 341–353.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of Hidden Field Equations (HFE) using Gröbner bases. In *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Dan Boneh, editor, Springer, 2003.

[Ful89] William Fulton. *Algebraic curves. An introduction to algebraic geometry, a Reprint of 1969 original in Advanced Book Classics.* Addison-Wesley Publishing Company, Redwood City, CA, 1989. ISBN: 0-201-51010-3.

[FW02] Patrick Fitzpatrick and Christopher Wolf. Direct division in factor rings. *Electronic Letters*, 38(21):1253–1254, October 2002. Extended version: `http://eprint.iacr.org/2004/353`, 7 pages.

[GC00] Louis Goubin and Nicolas T. Courtois. Cryptanalysis of the TTM cryptosystem. In *Advances in Cryptology — ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 44–57. Tatsuaki Okamoto, editor, Springer, 2000.

[IM85] Hideki Imai and Tsutomu Matsumoto. Algebraic methods for constructing asymmetric cryptosystems. In *Algebraic Algorithms and Error-Correcting Codes, 3rd International Conference, AAECC-3, Grenoble, France, July 15-19, 1985, Proceedings*, volume 229 of *Lecture Notes in Computer Science*, pages 108–119. Jacques Calmet, editor, Springer, 1985.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes. In *Advances in Cryptology — EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222. Jacques Stern, editor, Springer, 1999.

[LD00] Julio Lopéz and Ricardo Dahab. An overview of elliptic curve cryptography. Technical report, Institute of Computing, State University of Campinas, Brazil, 22nd of May 2000. `http://citeseer.nj.nec.com/333066.html` or `http://www.dcc.unicamp.br/ic-tr-ftp/2000/00-14.ps.gz`.

[MAG] Computational Algebra Group, University of Sydney. *The MAGMA Computational Algebra System for Algebra, Number Theory and Geometry.* http://magma.maths.usyd.edu.au/magma/.

[NES] NESSIE: New European Schemes for Signatures, Integrity, and Encryption. Information Society Technologies programme of the European commission (IST-1999-12324). http://www.cryptonessie.org/.

[Pat95] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88. In *Advances in Cryptology — CRYPTO 1995*, volume 963 of *Lecture Notes in Computer Science*, pages 248–261. Don Coppersmith, editor, Springer, 1995.

[Pat96] Jacques Patarin. Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology — EUROCRYPT 1996*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Ueli Maurer, editor, Springer, 1996. Extended Version: http://www.minrank.org/hfe.pdf.

[Pat97] Jacques Patarin. The oil and vinegar signature scheme. presented at the Dagstuhl Workshop on Cryptography, September 1997. transparencies.

[PKC05] Serge Vaudenay, editor. *Public Key Cryptography — PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*. Springer, 2005. ISBN 3-540-24454-9.

[Sha93] Adi Shamir. Efficient signature schemes based on birational permutations. In Cr [Cr93], pages 1–12.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.

[TKI+86] S. Tsujii, K. Kurosawa, T. Itoh, A. Fujioka, and T. Matsumoto. A public key cryptosystem based on the difficulty of solving a system of nonlinear equations. *ICICE Transactions (D) J69-D*, 12:1963–1970, 1986.

[WBP04] Christopher Wolf, An Braeken, and Bart Preneel. Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC. In *Conference on Security in Communication Networks — SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 294–309. Springer, September 8–10 2004. Extended version: http://eprint.iacr.org/2004/237.

[WHL+05] Lih-Chung Wang, Yuh-Hua Hu, Feipei Lai, Chun-Yen Chou, and Bo-Yin Yang. Tractable rational map signature. In PKC [PKC05], pages 244–257.

[WP04] Christopher Wolf and Bart Preneel. Asymmetric cryptography: Hidden Field Equations. In *European Congress on Computational Methods in Applied Sciences and Engineering 2004*. P. Neittaanmäki, T. Rossi, S. Korotov, E. Oñate, J. Périaux, and D. Knörzer, editors, Jyväskylä University, 2004. 20 pages, extended version: http://eprint.iacr.org/2004/072/.

[WP05a] Christopher Wolf and Bart Preneel. Equivalent keys in HFE, C*, and variations. In *Proceedings of Mycrypt 2005*, volume 3715 of *Lecture Notes in Computer Science*, pages

33–49. Serge Vaudenay, editor, Springer, 2005. Extended version `http://eprint.iacr.org/2004/360/`, 15 pages.

[WP05b] Christopher Wolf and Bart Preneel. Taxonomy of public key schemes based on the problem of multivariate quadratic equations. Cryptology ePrint Archive `http://eprint.iacr.org`, Report 2005/077, 12th of May 2005. `http://eprint.iacr.org/2005/077/`, 64 pages.

[WYHL06] Lih-Chung Wang, Bo-Yin Yang, Yuh-Hua Hu, and Feipei Lai. A "medium-field" multivariate public-key encryption scheme. In *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 132–149. David Pointcheval, editor, Springer, 2006.

[YC04a] Bo-Yin Yang and Jiun-Ming Chen. All in the XL family: Theory and practice. In *ICISC 2004*, pages 67–86. Springer, 2004.

[YC04b] Bo-Yin Yang and Jiun-Ming Chen. Theoretical analysis of XL over small fields. In *ACISP 2004*, volume 3108 of *LNCS*, pages 277–288. Springer, 2004.

[YC05] Bo-Yin Yang and Jiun-Ming Chen. Building secure tame-like multivariate public-key cryptosystems: The new TTS. In *ACISP 2005*, volume 3574 of *LNCS*, pages 518–531. Springer, July 2005.

A Extensions

The trapdoor ℓIC we described in this article can be extended in several ways. Due to efficiency considerations, we only discuss extensions which allow a public key $\mathcal{P}$ of maximal degree 2, *i.e.*, we discard constructions of cubic or higher type.

Even with this restriction, there are two obvious extensions. For the first we observe that the private key matrices are of degree $2k$. As this allows rank attacks (see Sec. 3.2), we would like to obtain a higher rank for the private key matrices. We can do this by allowing more input variables A_i per coordinate B_i with $1 \le i \le \ell$. In its most general form, we can associate to each coordinate B_i a set of quintuples from the following set:

$$\mathcal{S} \;:=\; \{(i,j,\lambda,\kappa,\gamma) \in \mathbb{N}^4 \times \mathbb{E}^* \mid 1 \le i \le j \le \ell \wedge 0 \le \lambda,\kappa < k\} \tag{7}$$

We now have an ℓ-tuple of sets $\mathcal{B}_i \subset \mathcal{S}$ for each output B_i:

$$B_i \;:=\; \sum_{(i,j,\lambda,\kappa,\gamma)\in\mathcal{B}_i} \gamma A_i^{q^\lambda} A_j^{q^\kappa} \quad \text{for } 1 \le i \le \ell \tag{8}$$

The challenge is to find sets $\mathcal{B}_i$, safe against all known attacks but still allowing easy inversion.

More Complicated Structures To effect inversion with product-type structures like in ℓIC, where each equation only has two variables, we *will* have a cycle-like structure like somewhere. This shows that the ℓIC is in some sense a basic trapdoor as we claimed.

Double cycles. A solution for the first is the Double Cycle paradigm for $\ell > 3$:

$$\mathcal{B}_i := \{(i,\mu(i),0,0,1),(i,\mu(\mu(i)),0,0,1)\} \text{ for } 1 \le i \le \ell \tag{9}$$

Note that we have $0 \in \mathbb{N}$ but $1 \in \mathbb{E}^*$ here. Unfortunately, these formulæ do not permit an easy closed inversion although they offer more protection against rank attacks as ℓIC polynomials do.

HFE-like polynomials. The second variation is in the spirit of the HFE system: instead of having monomials, we allow polynomials in the two input variables $A_i A_{\mu(i)}$ for each coordinate B_i and $1 \leq i \leq \ell$. More formally, we have

$$\mathcal{B}_i := \{(i, \mu(i), \lambda, \kappa, \gamma)\} \text{ for } 1 \leq i \leq \ell \tag{10}$$

To allow fast inversion, we will in most cases have some maximal degree $d \in \mathbb{N}$ and hence $q^\lambda + q^\kappa \leq d$ as additional condition in the sets from (10). However, the fact that we have polynomials instead of monomials will defeat certain attacks. In particular, Gröbner base algorithms (see Sec. 3.3) and Patarin-type attacks (see Sec. 3.1) are more difficult now. Obviously, it is possible to combine both idea and hence have high-rank HFE-like systems.

All trapdoors of the form (8) lead to $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key systems. However, inversion becomes more costly so we have to leave it as an open question if these schemes have advantages in practice.

B More About Gröbner Basis Attacks

When attacking with FXL or any more advanced algorithms of the Lazard-Faugeère family, and when the resulting matrix equation is solved in the same order as the Lanczos/Wiedemann type:

$$\min\left\{ q^f \binom{n+d-f}{d}^2 \left(c_0 + c_1 \lg \binom{n+d-f}{d} \right) \middle| \, d := \min\left\{ D : [t^D] \left(\frac{(1-t^2)^m (1-t^q)^{n-f}}{(1-t)^{n+1-f}(1-t^{2q})^m} \right) < 0 \right\} \right\}$$

The analysis is extremely complex. Bardet *et al* investigated the issue found the asymptotic estimates for the degree [BFS04, BFSY05, YC04b]. It is shown by Yang and Chen that for generic systems some measure of guessing to get overdeterminedness is always correct [YC04a]. The current consensus is that for $m = n = 20$ and $q = 256$, the complexity of solving a generic $\mathcal{MQ}$ instance is anywhere between 2^{72} to 2^{80} 3DES blocks using guessing and sparse matrices. With parameters in this heneral range, for every 4 variables and equations added, the complexity goes up by $\sim 2^{10}$. As $m = n$ increases, the best-case generic solving using a Lazard-Faugère type solver with optimized guessing goes up with $2^{2.4n}$ times a polynomial. For $q = 2$, the best-case timing for generic polynomials should go up with $2^{0.785n}$ times a polynomial.

C The Space of Linearization Equations

To know if the Patarin attack from Sec. 3.1 is efficient, we need to know the number of linearization equations. In particular, if this number is higher than expected, this attack becomes more efficient.

To simplify the proof, we restrict to $\ell = 2, q = 2$. However, the ideas presented in this section can be easily extended although the overall proof becomes more technical. As usual, we have $\mathbb{E}$ the ℓ^{th} degree extension of $\mathbb{F}$.

Let P be the ℓIC-mapping, *i.e.*, a map $\mathbb{E} \times \mathbb{E}$ to itself with

$$(B_1, B_2) := P(A_1, A_2) = (A_1 A_2, A_2 A_1^q)$$

The quadruple (B_1, B_2, A_1, A_2) forms a surface in $\mathbb{E}^4$. We are interested in finding the linearization equations over $\mathbb{F}$.

There are four obvious types:

$$B_1^q A_1 + B_2 A_1^q = 0, B_1 A_2^q + B_2 A_2 = 0, B_2 A_1 + B_1^q = 0, B_1 A_2 + B_2 = 0. \tag{11}$$

We now want to check if there are any other linearization equations then the ones coming from the linear combination of these four types. As we will see below, the answer is no.

PROPOSITION C.1 *For $q = 2$, any linearization equation comes from a linear combination of (11).*

PROOF. If there is any linearization equation, we know from [DGS+05] that it must come from a nontrivial equation of the form:

$$\begin{aligned} \sum_{i,j=0}^{\ell-1} a_{1,i,j} B_1^{q^i} A_1^{q^j} + \sum_{i,j=0}^{\ell-1} b_{1,i,j} B_1^{q^i} A_2^{q^j} + \sum_{i,j=0}^{\ell-1} a'_{2,i,j} B_2^{q^i} A_1^{q^j} \\ + \sum_{i,j=0}^{\ell-1} b'_{2,i,j} B_2^{q^i} A_2^{q^j} + \sum_{i=0}^{\ell-1} c_i A_1^{q^i} + \sum_{j=0}^{\ell-1} d_j A_2^{q^j} + \sum_{i=0}^{\ell-1} e_i B_1^{q^i} + \sum_{i=0}^{\ell-1} f_i B_2^{q^i} + g &= 0. \end{aligned}$$

For $a_{1,i,j}, b_{1,i,j}, a'_{2,i,j}, b'_{2,i,j}, c_i, d_j, e_i, f_i, g \in \mathrm{GF}(2)$ with $0 \le i, j < \ell$. The above equation can be written as

$$\begin{aligned} \sum_{i,j=0}^{\ell-1} a_{1,i,j} (A_1 A_2)^{q^i} A_1^{q^j} + \sum_{i,j=0}^{\ell-1} b_{1,i,j} (A_1 A_2)^{q^i} A_2^{q^j} + \sum_{i,j=0}^{\ell-1} a'_{2,i,j} (A_1 A_2^q)^{q^i} A_1^{q^j} \\ + \sum_{i,j=0}^{\ell-1} b'_{2,i,j} (A_1 A_2^q)^{q^i} A_2^{q^j} + \sum_{i=0}^{\ell-1} c_i A_1^{q^i} + \sum_{j=0}^{\ell-1} d_j A_2^{q^j} + \sum_{i=0}^{\ell-1} e_i (A_1 A_2)^{q^i} + \sum_{i=0}^{\ell-1} f_i (A_1 A_2^q)^{q^i} + g &= 0, \end{aligned}$$

for any pair (A_1, A_2). This implies $C_i = 0, d_j = 0$ for $0 \le i, j < \ell$ and $g = 0$. Therefore this equation simplifies to

$$\begin{aligned} \sum_{i,j=0}^{\ell-1} a_{1,i,j} A_1^{q^i+q^j} A_2^{q^i} + \sum_{i,j=0}^{\ell-1} b_{1,i,j} A_1^{q^i} A_2^{q^i+q^j} + \sum_{i,j=0}^{\ell-1} a'_{2,i,j} A_1^{q^i+q^j} A_2^{q^{i+1}} + \\ \sum_{i,j=0}^{\ell-1} b'_{2,i,j} A_1^{q^i} A_2^{q^{i+1}+q^j} + \sum_{i=0}^{\ell-1} e_i A_1^{q^i} A_2^{q^i} + \sum_{i=0}^{\ell-1} f_i A_1^{q^i} A_2^{q^{i+1}} &= 0. \end{aligned}$$

It is clear that

1. if $i \neq j$ we have $a_{1,i,j} = 0$, $b_{1,i,j} = 0$ and $a'_{2,i,j} = 0$ except for $a'_{1,i-1,i}, a_{1,i,i-1}$ because $q^i + q^j$ is not a power of q
2. if $i + 1 \neq j$ we have $b'_{2,i,j} = 0$, except for $b'_{2,i,i}, b_{1,i,i+1}$ because $q^{i+1} + q^j$ is not a power of q.

Moreover, we have

$$\begin{aligned}\sum_{i,j=0}^{\ell-1} a_{1,i,i-1}A_2^{q^i}A_1^{q^i+q^{i-1}} + \sum_{i,j=0}^{\ell-1} a'_{1,i-1,i}A_2^{q^{i-1}}A_1^{q^i+q^{i-1}} + \\ \sum_{i,j=0}^{\ell-1} b_{1,i,i+1}A_1q^iA_2^{q^i+q^{i+1}} + \sum_{i,j=0}^{\ell-1} b'_{2,i,i}A_1^{q^i}A_2^{q^i+q^{i+1}} + \\ \sum_{i,j=0}^{\ell-1} a_{1,i,i}A_1^{q^{i+1}}A_2^{q^i} + \sum_{i,j=0}^{\ell-1} b_{1,i,i}A_1^{q^i}A_2^{q^{i+1}} + \\ \sum_{i,j=0}^{\ell-1} a'_{2,i,i}A_1^{q^{i+1}}A_2^{q^{i+1}} + \sum_{i,j=0}^{\ell-1} b'_{2i,i+1}A_1^{q^i}A_2^{q^{i+2}} + \\ \sum_{i=0}^{\ell-1} e_iA_1^{q^i}A_2^{q^i} + \sum_{i=0}^{\ell-1} f_iA_1^{q^i}A_2^{q^{i+1}} &= 0.\end{aligned}$$

This implies

$$b'_{2,i,i} = b_{1,i,i+1}, a'_{1,i-1,i} = a_{1,i,i-1}, a_{1,i,i} = b'_{2,i,i+1} = 0, a'_{2,i,i} = e_{i+1}, b_{1,i,i} = f_i.$$

In other words, all linearization equations come from the following form:

$$\begin{aligned}\sum_{i,j=0}^{\ell-1} a_{1,i,i-1}B_1^{q^i}A_1^{q^{i-1}} + \sum_{i,j=0}^{\ell-1} a_{1,i,i-1}B_2^{q^{i-1}}A_1^{q^i} + \\ \sum_{i,j=0}^{\ell-1} b_{1,i,i+1}B_1^{q^i}A_2^{q^{i+1}} + \sum_{i,j=0}^{\ell-1} b_{1,i,i+1}B_2^{q^i}A_2^{q^i} + \\ \sum_{i=0}^{\ell-1} e_iB_2^{q^{i-1}}A_1^{q^{i-1}} + \sum_{i=0}^{\ell-1} e_iB_1^{q^i} + \\ \sum_{i=0}^{\ell-1} f_iB_1^{q^i}A_2^{q^i} + \sum_{i=0}^{\ell-1} f_iB_2^{q^i} &= 0\end{aligned}$$

This can be rewritten as

$$\begin{aligned}\sum_{i,j=0}^{\ell-1} a_{1,i,i-1}(z_1^qA_1 + z_2A_1^q)^{q^{i-1}} + \sum_{i,j=0}^{\ell-1} b_{1,i,i+1}(z_1A_2^q + z_2A_2)^{q^i} + \\ \sum_{i,j=0}^{\ell-1} e_i(B_2A_1 + B_1^q)^{q^i} + \sum_{i,j=0}^{\ell-1} f_i(B_1A_2 + B_2)^{q^i} &= 0.\end{aligned}$$

□

Similarly we can show that the dimension of the space of linearization equations is exactly 4ℓ. We also see that if we substitute values for Z_1, Z_2, we can solve for A_1, A_2. In particular this means that linear equations derived for given ciphertext is of full rank.

D Choices of Parameters

We investigate several choices for a signature scheme based on ℓIC- with given security level. Aim is to find an optimal point in terms of efficiency and key size. The data is too long to give here and fit within a reasonable limit, but the conclusion is: $\ell = 3$ is best for signature and $\ell = 2$ for encryption (with a "plus" modifier).

More quantum algorithms

Oded Regev

Tel-Aviv University

In this talk I will describe some attempts to construct quantum algorithms to deal with problems not addressed by Shor's algorithms and subsequent developments. A particular emphasis will be put on quantum algorithms for lattice problems, as those are one of our best candidates for post-quantum cryptography. Among other things, I will describe the connection to the dihedral hidden subgroup problem, and show how the ability to create certain quantum states implies interesting quantum algorithms.

High Order Linearization Equation (HOLE) Attack on Multivariate Public Key Cryptosystems

Jintai Ding[1], Lei Hu[2], Xuyun Nie[2], Jianyu Li[2], John Wagner[1]

[1] Department of Mathematical Sciences,
University of Cincinnati,
Cincinnati, OH, 45220, USA
[2] State Key Laboratory of Information Security,
Graduate School of Chinese Academy of Sciences,
Beijing 100049, China
ding@math.uc.edu, {hu, nxy04b, ljy}@is.ac.cn, wagnerjh@email.uc.edu

Abstract. In the CT-track of the 2006 RSA conference, a new multivariate public key cryptosystem, which is called the Medium Field Equation (MFE) multivariate public key cryptosystem, is proposed by Wang, Yang, Hu and Lai. We use the second order linearization equation attack method by Patarin to break MFE. Given a ciphertext, we can derive the plaintext within 2^{23} $\mathbb{F}_{2^{16}}$-operations, after performing once for any public key a computation of complexity less than 2^{52}. We also propose a high order linearization equation (HOLE) attack on multivariate public key cryptosystems, which is a further generalization of the (first and second order) linearization equation (LE). This method can be used to attack extensions of the current MFE.

Keywords: multivariate public key cryptosystem, quadratic polynomial, algebraic cryptanalysis, high order linearization equation.

1 Introduction

For the last three decades, public key cryptosystems, as a revolutionary breakthrough in cryptography, have developed into an indispensable element of our modern communication system. For RSA and other number theory based cryptosystems, their security depends on the assumption about the difficulty of certain number theory problems, such as the Integer Prime Factorization Problem or the Discrete Logarithm Problem. However, due to the quantum computer attack by Shor [Sho99] and the demand for more efficient cryptosystems for small devices, there is a great challenge to build new public key cryptosystems, in particular ones that could survive future attacks utilizing quantum computers [PQ].

One such research direction utilizes a set of multivariate polynomials over a finite field, in particular, quadratic polynomials, as the public key of the cipher, which are called multivariate public key cryptosystems (MPKC). This method

is based on the proven theorem that solving a set of multivariate quadratic polynomial equations over a finite field generally is an NP-hard problem. Note, however, this does not guarantee that these new cryptosystems are secure. In the last decade, there has been tremendous amount of work devoted to this area. In 2004, one such cryptosystem, Sflash [ACDG03] [PCG01a], was accepted as one of the final selections in the New European Schemes for Signatures, Integrity, and Encryption: IST-1999-12324. A more efficient family of Rainbow signature schemes was also proposed last year [DS05] [YC05] [WHLCY05].

In the development of MPKC, one particular interesting and important new area is the development of the so-called algebraic attack. This new attack method started from the linearization equation (LE) attack by Patarin [Pat95], which is used to break Matsumoto-Imai cryptosystems. A linearization equation is an equation in the form:

$$\sum a_{ij}u_iv_j + \sum b_iu_i + \sum c_jv_j + d = 0,$$

where the u_i are components of the plaintext and the v_j are components of the ciphertext.

Later, Patarin, Courtois, Shamir, and Kipnis generalized this method by multiplying high order terms $u_1^{\alpha_1}\cdots u_n^{\alpha_n}$ of the plaintext variables but using only linear terms of ciphertext variables (v_j), which is called the XL method [CKPS00]. The method is closely related to the new Gröbner basis method by Faugere [Fau99] [AFIKS04]. Furthermore, this new algebraic method was used to attack symmetric ciphers like AES and others [CPi02]. One can see that algebraic attacks are becoming increasingly important in cryptography.

Another generalization of LE also by Patarin[Pat96,PCG01a,C00], which is not as well-known, is the type of equations in the form:

$$\sum a_{ijk}u_iv_jv_k + \sum b_{ij}u_iv_j + \sum c_iu_i + \sum d_{jk}v_jv_k + \sum e_jv_j + f = 0.$$

As a further extension, we propose to call the equations that that use high order terms of the ciphertext variables (v_j) while using only linear terms of plaintext variables (u_i), high order linearization equations (HOLE). The total degree of the highest order of the ciphertext variables (v_j) is called the order of the HOLE and the equation above is thus called a second order linearization equation (SOLE). For any MPKC, if we can derive such equations, then for any given ciphertext, we can insert it into the HOLEs, producing linear equations satisfied by the plaintext and these equations can be used to attack the system.

It turns out that the SOLEs can be used efficiently to break the Medium Field Equation (MFE) multivariate public key cryptosystem proposed by Wang, Yang, Hu and Lai in the CT-track of the 2006 RSA conference [WYH06].

MFE is an encryption scheme. Many encryption schemes of MPKC have been proposed, and many of them have been broken, for example, the TTM cryptosystem family [Moh99] [GC00] [CM01] [DS03a] [DS03b] [MCY04]. A very different direction goes along the idea started by Matsumoto and Imai [MI88], which can be generally called the "Big Field" idea.

Given a multivariate public key cryptosystem, the public key is defined as a map over the vector space $\mathbb{K}^n$, where $\mathbb{K}$ is a small finite field with q elements. However from the theory of finite fields, $\mathbb{K}^n$ can also be identified with a "big" finite field $\mathbb{E}$, which is a degree n extension of $\mathbb{K}$. That is, there is a standard $\mathbb{K}$-linear vector space isomorphism that identifies $\mathbb{E}$ with $\mathbb{K}^n$. The idea of the "Big Field" is that we can find a map, say ϕ_2, that is easy to invert on $\mathbb{E}$. Under the isomorphism we can build a map $\tilde{\phi_2}$: $\mathbb{K}^n \to \mathbb{K}^n$ as:

$$\tilde{\phi_2}(u_1, ..., u_n) \mapsto (g_1(u_1, ..., u_n), \cdots, g_n(u_1, ..., x_n)).$$

Then we use ϕ_1 and ϕ_3, two randomly chosen invertible affine linear maps over $\mathbb{K}^n$ which are the key part of the private key to "hide" ϕ_2. The public key is given by

$$\begin{aligned} &\bar{\phi_2}(u_1, ..., u_n) = \phi_3 \circ \tilde{\phi_2} \circ \phi_1(u_1, ..., u_n) \\ &= (h_1(u_1, ..., u_n), h_2(u_1, ..., u_n), \cdots, h_n(u_1, ..., u_n)). \end{aligned}$$

The Matsumoto-Imai (MI) cryptosystem was broken by Patarin [Pat95], and later Patarin developed the HFE cryptosystem [Pat96]. The only difference between HFE and the MI is that they choose different ϕ_2. Currently the more promising cryptosystems are new variants of the MI and the HFE through Oil-Vinegar constructions and internal perturbations [Din04a] [FGS05] [DG05] [DS04a]. The idea to put several "big fields" together to build a cryptosystem is also used [MI88] [Pat96]. The new MFE cryptosystem [WYH06] uses what the designers call "Medium Field Encryption". The non-linear critical part of the public key is a function over an extension of the base field $\mathbb{K}$ of degree smaller than what would be called the "big field". Another key difference between MFE and HFE is that MFE uses functions derived from a matrix structure while the MI and the HFE use only polynomials of a single variable.

In the attack on MFE, we first use second order linearization equations (SOLEs), which we derive from the special algebraic structure of the crucial nonlinear map in MFE. This is the most essential step in our attack. Any given ciphertext can be inserted into the SOLEs to produce a set of equations linear in the plaintext variables. Solutions to these equations are finally plugged back into the original public key polynomial equations, providing a set of new quadratic equations that could be easily solved. The complexity of our break is less than 2^{52} one-time computations over $\mathbb{K}$ for any given public key, and the practical complexity of recovering a ciphertext is less than 2^{23} $\mathbb{K}$-operations.

The current MFE is based on matrices of size 2×2 and one may extend it to a construction using matrices of bigger size. The HOLEs of higher order can be extended to attack such an extension of the current MFE and the order of HOLE corresponds exactly to the size of the matrices.

We organize the paper as follows. We introduce the MFE cryptosystem in Section 2, and present our attack in Section 3. In Section 4, we discuss the connection of HOLE with the XL method. In the final section, we present the conclusion.

2 MFE Public Key Cryptosystem

Let $\mathbb{K}$ be a finite field, generally $\mathbb{F}_{2^{16}}$. Let $\mathbb{L}$ be its degree r extension field; $\mathbb{L}$ is considered the "Medium Field".

In MFE, we always identify $\mathbb{L}$ with $\mathbb{K}^r$ by a $\mathbb{K}$-linear isomorphism $\pi : \mathbb{L} \to \mathbb{K}^r$. Namely we take a basis of $\mathbb{L}$ over $\mathbb{K}$, $\{\theta_1, \cdots, \theta_r\}$, and define π by $\pi(a_1\theta_1 + \cdots + a_r\theta_r) = (a_1, \cdots, a_r)$ for any $a_1, \cdots, a_r \in \mathbb{K}$. It is natural to extend π to two $\mathbb{K}$-linear isomorphisms $\pi_1 : \mathbb{L}^{12} \to \mathbb{K}^{12r}$ and $\pi_2 : \mathbb{L}^{15} \to \mathbb{K}^{15r}$.

A private key of MFE consists of two invertible linear affine transformations ϕ_1 and ϕ_3; and ϕ_1 is defined on $\mathbb{K}^{12r}$, and ϕ_3 on $\mathbb{K}^{15r}$. Let $\phi_2 : \mathbb{L}^{12} \to \mathbb{L}^{15}$ be the central nonlinear quadratic map of MFE. Note ϕ_2 is fixed except for the three components Q_1, Q_2, and Q_3, which have randmly chosen coefficients. The corresponding public key is $15r$ quadratic polynomials $h_1(u_1, ..., u_{12r}), h_2(u_1, ..., u_{12r}), \cdots$, and $h_{15r}(u_1, ..., u_{12r})$ given by

$$(h_1(u_1, ..., u_{12r}), \cdots, h_{15r}(u_1, ..., u_{12r})) = \phi_3 \circ \pi_2 \circ \phi_2 \circ \pi_1^{-1} \circ \phi_1(u_1, ..., u_{12r}). \quad (1)$$

Let $\phi_2(X_1, \cdots, X_{12}) = (Y_1, \cdots, Y_{15})$. The expressions of the Y_i are given by

$$\begin{cases} Y_1 = X_1 + X_5X_8 + X_6X_7 + Q_1; \\ Y_2 = X_2 + X_9X_{12} + X_{10}X_{11} + Q_2; \\ Y_3 = X_3 + X_1X_4 + X_2X_3 + Q_3; \\ Y_4 = X_1X_5 + X_2X_7; \qquad Y_5 = X_1X_6 + X_2X_8; \\ Y_6 = X_3X_5 + X_4X_7; \qquad Y_7 = X_3X_6 + X_4X_8; \\ Y_8 = X_1X_9 + X_2X_{11}; \qquad Y_9 = X_1X_{10} + X_2X_{12}; \\ Y_{10} = X_3X_9 + X_4X_{11}; \qquad Y_{11} = X_3X_{10} + X_4X_{12}; \\ Y_{12} = X_5X_9 + X_7X_{11}; \qquad Y_{13} = X_5X_{10} + X_7X_{12}; \\ Y_{14} = X_6X_9 + X_8X_{11}; \qquad Y_{15} = X_6X_{10} + X_8X_{12}. \end{cases} \quad (2)$$

Here Q_1, Q_2, and Q_3 form a triple (Q_1, Q_2, Q_3) which is a triangular map from $\mathbb{K}^{3r}$ to itself as follows. Let $\pi(X_1) = (x_1, \cdots, x_r)$, $\pi(X_2) = (x_{r+1}, \cdots, x_{2r})$, $\pi(X_3) = (x_{2r+1}, \cdots, x_{3r})$, and let $q_i \in \mathbb{K}[x_1, \cdots, x_{i-1}]$ for $2 \le i \le 3r$. Then

$$\begin{cases} Q_1(X_1) = \sum_{i=2}^{r} q_i(x_1, \cdots, x_{i-1})\theta_i, \\ Q_2(X_1, X_2) = \sum_{i=r+1}^{2r} q_i(x_1, \cdots, x_{i-1})\theta_i, \\ Q_3(X_1, X_2, X_3) = \sum_{i=2r+1}^{3r} q_i(x_1, \cdots, x_{i-1})\theta_i. \end{cases}$$

The q_i can be any randomly chosen quadratic polynomials. A specific "tower"-structural choice for them is given in §5 of [WYH06].

The encryption of MFE is the evaluation of public key polynomials, namely given a plaintext $(u_1, \cdots, u_{12r})$, its ciphertext is

$$(v_1, \cdots, v_{15r}) = (h_1(u_1, \cdots, u_{12r}), \cdots, h_{15r}(u_1, \cdots, u_{12r})).$$

Given a valid ciphertext $(v_1, \cdots, v_{15r})$, the decryption of MFE is to calculate in turn $\phi_1^{-1} \circ \pi_1 \circ \phi_2^{-1} \circ \pi_2^{-1} \circ \phi_3^{-1}(v_1, \cdots, v_{15r})$. Here the point is how to invert ϕ_2, its basic idea is to use the triangular structure of ϕ_2. Relating to our cryptanalysis, the method of computing ϕ_2^{-1} is listed as follows, see §4.2 and Appendix B of [WYH06].

Write $X_1, \cdots, X_{12}, Y_4, \cdots, Y_{15}$ as six 2×2 matrices:

$$\begin{aligned} &M_1 = \begin{pmatrix} X_1 & X_2 \\ X_3 & X_4 \end{pmatrix}, M_2 = \begin{pmatrix} X_5 & X_6 \\ X_7 & X_8 \end{pmatrix}, M_3 = \begin{pmatrix} X_9 & X_{10} \\ X_{11} & X_{12} \end{pmatrix}, \\ &Z_3 = M_1 M_2 = \begin{pmatrix} Y_4 & Y_5 \\ Y_6 & Y_7 \end{pmatrix}, Z_2 = M_1 M_3 = \begin{pmatrix} Y_8 & Y_9 \\ Y_{10} & Y_{11} \end{pmatrix}, \\ &Z_1 = M_2^T M_3 = \begin{pmatrix} Y_{12} & Y_{13} \\ Y_{14} & Y_{15} \end{pmatrix}. \end{aligned} \tag{3}$$

Then

$$\begin{cases} \det(M_1) \cdot \det(M_2) = \det(Z_3), \\ \det(M_1) \cdot \det(M_3) = \det(Z_2), \\ \det(M_2) \cdot \det(M_3) = \det(Z_1). \end{cases}$$

When M_1, M_2, and M_3 are all invertible, we can get values of $\det(M_1)$, $\det(M_2)$, and $\det(M_3)$ from $\det(Z_1)$, $\det(Z_2)$, and $\det(Z_3)$, for instance, $\det(M_1) = \big(\det(Z_2) \cdot \det(Z_3)/\det(Z_1)\big)^{1/2}$. The square root operation is easy to handle over a field of characteristic 2.

With values of $\det(M_1)$, $\det(M_2)$, and $\det(M_3)$, we solve the following triangular map over $\mathbb{K}^{3r}$

$$\begin{cases} Y_1 = X_1 + Q_1 + \det(M_2) \\ Y_2 = X_2 + Q_2 + \det(M_3) \\ Y_3 = X_3 + Q_3 + \det(M_1) \end{cases} \tag{4}$$

to get in turn $x_1, \cdots, x_r, x_{r+1}, \cdots, x_{2r}, x_{2r+1}, \cdots$, and x_{3r}. Thus, we recover X_1, X_2, and X_3. From $X_1X_4 + X_2X_3 = \det(M_1)$ we then get X_4 provided $X_1 \neq 0$. The $X_5, \cdots, X_{12}$ are consequently solved from the 4th to 11th equations of (2). Appendix B of [WYH06] presents the method of computing the X_i in the case when $X_1 = 0$. It is slightly easier than the case of $X_1 \neq 0$.

If there is a non-invertible matrix among M_1, M_2, and M_3, then the decryption mentioned above will not work. This decryption failure exists in MFE [WYH06]. We call a plaintext singular if its corresponding M_1, M_2, and M_3 are not all invertible, otherwise it is called **nonsingular**. The ciphertext of a nonsingular plaintext is called a nonsingular ciphertext.

It is easy to prove that the ratio of singular plaintexts among all plaintexts is at most $4|\mathbb{L}|^{-1}$; when $\mathbb{L} = \mathbb{F}_{2^{64}}$, the ratio is at most 2^{-62} and is very small. See Appendix A. In the next section we only consider how to recover nonsingular ciphertext.

There are two typical instances of MFE proposed by the designers of MFE.

1) MFE-1, where $\mathbb{K} = \mathbb{F}_{2^{16}}$ and $r = 4$. The public key has 60 polynomials with 48 variables.

2) MFE-1′, where $\mathbb{K} = \mathbb{F}_{2^{16}}$ and $r = 5$. The public key has 75 polynomials and 60 variables.

There is also a mini-version of MFE (MFE-0) using $\mathbb{K} = \mathbb{F}_{2^8}$ and $r = 4$, which has the same number of polynomials and variables as MFE-1.

3 Cryptanalysis on MFE

The designers of MFE noted they should avoid the linearization attack of Patarin (§6.2 of [WYH06]), and this is indeed the case. In the design of MFE, the last equations of (2) in MFE is defined such that $Z_1 = M_2^T M_3$ (see (2)), not $Z_1 = M_2 M_3$, otherwise we would have $Z_3 M_3 = M_1 Z_1$ $(= M_1 M_2 M_3)$, and this would have produced linearization equations for the cryptosystem. However we can use the HOLE, in particular the SOLE, to attack this cryptosystem.

3.1 Second Order Linearization Equations

First, we will show algebraically why the MFE has second order linearization equations.

Denote by M^* the associated matrix of a square matrix; for $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, its associated matrix is $M^* = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$. From (3), we have

$$Z_3 = M_1 M_2, \qquad Z_2 = M_1 M_3. \tag{5}$$

From these, we can derive

$$M_3 M_3^* M_1^* M_1 M_2 = M_3 (M_1 M_3)^* (M_1 M_2) = M_3 Z_2^* Z_3,$$

$$M_3 M_3^* M_1^* M_1 M_2 = (M_3 M_3^*)(M_1 M_1^*) M_2 = \det(M_3)\det(M_1) M_2 = \det(Z_2) M_2,$$

and hence,

$$M_3 Z_2^* Z_3 = \det(Z_2) M_2, \tag{6}$$

that is,

$$\begin{pmatrix} X_9 & X_{10} \\ X_{11} & X_{12} \end{pmatrix} \begin{pmatrix} Y_{11} & -Y_9 \\ -Y_{10} & Y_8 \end{pmatrix} \begin{pmatrix} Y_4 & Y_5 \\ Y_6 & Y_7 \end{pmatrix} = (Y_8 Y_{11} - Y_9 Y_{10}) \begin{pmatrix} X_5 & X_6 \\ X_7 & X_8 \end{pmatrix}. \tag{7}$$

Expanding (7), we get four equations of the form

$$\sum a'_{ijk} X_i Y_j Y_k = 0, \tag{8}$$

which hold for any corresponding pair $(X_1, \cdots, X_{12}, Y_1, \cdots, Y_{15})$. For any non-singular plaintext, if we substitute all the Y_i by its corresponding value in the four equations of the form (8) derived from (7), we would get four linear equations with X_i as its variables, and these four equations are linearly independent, since the matrices $\begin{pmatrix} Y_{11} & Y_9 \\ Y_{10} & Y_8 \end{pmatrix}$ and $\begin{pmatrix} Y_4 & Y_5 \\ Y_6 & Y_7 \end{pmatrix}$ are invertible.

Substituting $(X_1, \cdots, X_{12}) = \pi_1^{-1} \circ \phi_1(u_1, \cdots, u_{12r})$ and $(Y_1, \cdots, Y_{15}) = \pi_2^{-1} \circ \phi_3^{-1}(v_1, \cdots, v_{15r})$ into (8), we get $4r$ equations of the form

$$\sum_i u_i \left(\sum_{j \le k} a_{ijk} v_j v_k + \sum_j b_{ij} v_j + c_i \right) + \sum_{j \le k} d_{jk} v_j v_k + \sum_j e_j v_j + f = 0, \quad (9)$$

where the coefficients $a_{ijk}, b_{ij}, c_i, d_{jk}, e_j, f \in \mathbb{K}$, and the summations are respectively over $1 \le i \le 12r$, $1 \le j \le k \le 15r$ and $1 \le j \le 15r$. These equations, which are linear in plaintext components u_i and quadratic in ciphertext components v_j, are **second order linearization equations (SOLEs)**. It is easy to show that when all the v_j are substituted by any nonsingular ciphertext, the $4r$ SOLEs derived from (9) become linearly independent linear equations in u_i.

Similarly to (6), we can deduce from (5) another equation

$$M_2 Z_3^* Z_2 = \det(Z_3) M_3, \quad (10)$$

or in its matrix form,

$$\begin{pmatrix} X_5 & X_6 \\ X_7 & X_8 \end{pmatrix} \begin{pmatrix} Y_7 & -Y_5 \\ -Y_6 & Y_4 \end{pmatrix} \begin{pmatrix} Y_8 & Y_9 \\ Y_{10} & Y_{11} \end{pmatrix} = (Y_4 Y_7 - Y_5 Y_6) \begin{pmatrix} X_9 & X_{10} \\ X_{11} & X_{12} \end{pmatrix}. \quad (11)$$

The $4r$ SOLEs resulted from (11) are clearly different from the ones corresponding to (9). Furthermore, we can show that the $8r$ SOLEs obtained from (9) and (11) are all linearly independent. However, we note that when the v_i in these $8r$ SOLEs derived from (7) and (11) are assigned any nonsingular ciphertext, we will get only $4r$ linearly independent linear equations in u_i. In other words, once the values of v_i are given, as linear equations in X_i, (10) is completely equivalent to (6), and one can deduce (10) directly from (6) and vice versa. One can see this by the fact that multiplying from the right the both sides of (6) by $Z_3^* Z_2 / \det(Z_2)$ (this is a constant invertible matrix if the y_i values are given) gives (10).

Now, it is obvious that there are more SOLEs. We apply the above trick that results (6) and (10) from (5) to obtain

$$M_2 (Z_1^T)^* Z_2^T = \det(Z_1) M_1^T, \quad (12)$$

$$M_1^T (Z_2^T)^* Z_1^T = \det(Z_2) M_2, \quad (13)$$

from $Z_2 = M_1 M_3$ and $Z_1 = M_2^T M_3$. We can also obtain

$$M_1^T (Z_3^T)^* Z_1 = \det(Z_3) M_3, \quad (14)$$

$$M_3 (Z_1)^* Z_3^T = \det(Z_1) M_1^T, \quad (15)$$

from $Z_3 = M_1 M_2$ and $Z_1 = M_2^T M_3$. It is not hard to check that the polynomial equations derived from (6), (10), and (12)-(15) in terms of X_i and Y_j are all linearly independent. Thus, we get at least $24r$ linearly independent SOLEs in u_i and v_i over $\mathbb{K}$.

To find all SOLEs, we need to evaluate sufficiently many plain/cipher-texts in (9) to get a system of linear equations on the $a_{ijk}, b_{ij}, \cdots, f$. Let s be the dimension of its solution space and $(a_{ijk}^{(l)}, b_{ij}^{(l)}, \cdots, f^{(l)})$, $1 \leq l \leq s$, be its s linearly independent solutions. As mentioned above, we know $s \geq 24r$. For attack purposes, we only need to do the computation to get all the SOLEs once for any given public key.

Similarly to the relation between (6) and (10), as linear equations in X_i, (12) is equivalent to (13), and (14) is equivalent to (15) provided that the Y_i are assigned a nonsingular ciphertext value.

In addition, we can show that if we are given the values of v_i of a nonsingular ciphertext, from the $24r$ linearly independent SOLEs we derived above, we will produce only $8r$ linearly independent linear equations in u_i. Write (12) in its matrix form:

$$\begin{pmatrix} X_5 & X_6 \\ X_7 & X_8 \end{pmatrix} \begin{pmatrix} Y_{15} & -Y_{14} \\ -Y_{13} & Y_{12} \end{pmatrix} \begin{pmatrix} Y_8 & Y_{10} \\ Y_9 & Y_{11} \end{pmatrix} = (Y_{12}Y_{15} - Y_{13}Y_{14}) \begin{pmatrix} X_1 & X_3 \\ X_2 & X_4 \end{pmatrix}, \quad (16)$$

which results in $4r$ SOLEs. Given the values of Y_i of a nonsingular ciphertext, the eight linear equations in X_i derived from (16) and (7) are linearly independent, because the coefficient matrix corresponding to the set of eight linear equations, with the four equations from (16) as the first four ones, is in the form $\begin{pmatrix} I * 0 \\ 0\ I\ * \end{pmatrix}$, where each row is scaled by a factor $Y_8Y_{11} - Y_9Y_{10}$ or $Y_{12}Y_{15} - Y_{13}Y_{14}$ correspondingly, and I and 0 are respectively the identity matrix and the zero matrix of order 4. This matrix is clearly of rank 8. This shows that the s' introduced in the next subsection is at least $8r$. The reason that the other SOLEs will not produce any new linear equations on u_i for any given values of v_i of a nonsingular ciphertext is that when the Y_i are assigned a nonsingular value, (14) can be easily deduced from (6) and (12).

3.2 Ciphertext-only Attack

Now assume we have found a basis of the linear space of all SOLEs.

Given a ciphertext $(v'_1, \cdots, v'_{15r})$, our aim is to recover its plaintext $(u'_1, \cdots, u'_{12r})$.

We plug the values of ciphertext $(v'_1, \cdots, v'_{15r})$ into the basis SOLEs:

$$\begin{cases} \sum_i u_i \left(\sum_{j \leq k} a_{ijk}^{(l)} v'_j v'_k + \sum_j b_{ij}^{(l)} v'_j + c_i^{(l)} \right) + \sum_{j \leq k} d_{jk}^{(l)} v'_j v'_k + \sum_j e_j^{(l)} v'_j + f^{(l)} = 0 \\ 1 \leq l \leq s \end{cases} \quad (17)$$

giving us a linear system on $u_1, \cdots, u_{12r}$. Assume it has s' linearly independent solutions. From the previous subsection, we know $8r \leq s' \leq 12r$. We can represent s' of variables $u_1, \cdots, u_{12r}$ by linear affine expressions of the remaining $t := 12r - s'$. Let $w_1, \cdots, w_t$ be these t variables.

Substitute these s' linear expressions into the original public key polynomials to get $15r$ new quadratic polynomials $\widetilde{h_1}(w_1, ..., w_t), \widetilde{h_2}(w_1, ..., w_t), \cdots$, and $\widetilde{h_{15r}}(w_1, ..., w_t)$.

Let S be the solution space of (17). Let Y'_i and Z'_i be components and matrices corresponding to the given $(v'_1, \cdots, v'_{15r})$, namely

$$(Y'_1, \cdots, Y'_{15}) = \pi_2^{-1} \circ \phi_3^{-1}(v'_1, \cdots, v'_{15r}),$$

$$Z'_3 = \begin{pmatrix} Y'_4 & Y'_5 \\ Y'_6 & Y'_7 \end{pmatrix}, Z'_2 = \begin{pmatrix} Y'_8 & Y'_9 \\ Y'_{10} & Y'_{11} \end{pmatrix}, Z'_1 = \begin{pmatrix} Y'_{12} & Y'_{13} \\ Y'_{14} & Y'_{15} \end{pmatrix}.$$

We have found a basis of all SOLEs and each SOLE is a linear combination of this basis. This fact holds when the variables v_i in the equations are substituted by v'_i. Applying this fact to (7), we know the four resulting equations in u_i from

$$M_3(Z'_2)^* \cdot Z'_3 = \det(Z'_2) M_2 \tag{18}$$

are all linear combinations of the equations in (17). In other words, (18) holds on S. Let $P_{23} = \det(Z'_2)\left((Z'_2)^* \cdot Z'_3\right)^{-1}$; then $M_3 = M_2 P_{23}$. P_{23} is a constant matrix dependent only on the ciphertext.

Now we have that $M_2^T M_3 = Z_1$ always holds on $\mathbb{K}^{12r}$, therefore, we have that $M_3^T M_3 = M_3^T M_2 P_{23} = Z_1 P_{23}$ holds on S. That is,

$$\begin{pmatrix} X_9^2 + X_{11}^2 & X_9X_{10} + X_{11}X_{12} \\ X_9X_{10} + X_{11}X_{12} & X_{10}^2 + X_{12}^2 \end{pmatrix} = \begin{pmatrix} Y_{12} & Y_{13} \\ Y_{14} & Y_{15} \end{pmatrix} P_{23} \tag{19}$$

holds on S. Comparing the diagonal entries of the matrices in the both sides of (19), we find $X_9^2 + X_{11}^2$ and $X_{10}^2 + X_{12}^2$ are linear combinations of the Y_i. Applying ϕ_1 and ϕ_3 to these combinations and utilizing the fact that squaring is a linear operation on a field of characteristic 2, we have, on S, the $2r$ expressions corresponding to $X_9^2 + X_{11}^2$ and $X_{10}^2 + X_{12}^2$ are of the form $\sum a'_i u_i^2 + b'$ and $\mathbb{K}$-linear combinations of $h_1(u_1, ..., u_{12r}), h_2(u_1, ..., u_{12r}), \cdots, h_{15r}(u_1, ..., u_{12r})$ and 1 (constant).

Thus, of linear combinations of $\widetilde{h_1}(w_1, ..., w_t), \cdots, \widetilde{h_{15r}}(w_1, ..., w_t)$ and 1, there must exist $2r$ which contain only squaring terms and a constant term and correspond to $X_9^2 + X_{11}^2$ and $X_{10}^2 + X_{12}^2$.

It is easy to solve the following linear system on the $\widetilde{a_i}$ and $\widetilde{b_j}$:

$$\begin{cases} \sum\limits_{i=1}^{15r} \widetilde{a_i}\widetilde{h_i}(w_1, ..., w_t) + \sum\limits_{j=1}^{t} \widetilde{b_j} w_j^2 + \widetilde{c} = 0 \\ \forall w_1, ..., w_t \in \mathbb{K} \end{cases} \tag{20}$$

Essentially, this is to solve a linear equation system whose coefficients are the coefficients of the cross-terms and linear terms of the $\widetilde{h_i}(w_1, ..., w_t)$.

Let $(\widetilde{a_1}^{(l)}, \cdots, \widetilde{a_{15r}}^{(l)}, \widetilde{b_1}^{(l)}, \cdots, \widetilde{b_t}^{(l)})$, $1 \le l \le p$, be a basis of the solutions of (20). Set

$$\begin{cases} \sum\limits_{j=1}^{t} \left(\widetilde{b_j}^{(l)}\right)^{1/2} w_j + \left(\sum\limits_{i=1}^{15r} \widetilde{a_i}^{(l)} v'_i + \widetilde{c}^{(l)}\right)^{1/2} = 0 \\ 1 \le l \le p \end{cases} \tag{21}$$

For each $(u_1, ..., u_{12r}) \in S$, its corresponding $(w_1, ..., w_t)$ satisfies (21). From (21) we can represent p of the variables $w_1, ..., w_t$ by the remaining $t-p$ linearly. Totally, $s'+p$ components of the plaintext vector $(u'_1, ..., u'_{12r})$ are represented linearly by the remaining $12r - s' - p$.

Note that we surely have $s' + p \geq 10r$, since the matrix of the coefficients on $X_1, X_2, \cdots, X_{12}$ of ten expansions in (16), (7), $(X_9^2+X_{11}^2)^{1/2}$, and $(X_{10}^2+X_{12}^2)^{1/2}$ is $\begin{pmatrix} I * 0 \\ 0\ I\ * \\ 0\ 0\ A \end{pmatrix}$, where $A = \begin{pmatrix} 1\,0\,1\,0 \\ 0\,1\,0\,1 \end{pmatrix}$, and the matrix is obviously of rank 10. In other words, solving two systems (17) and (21) eliminates at least $10r$ variables of the plaintext components. If $p = 0$, i.e., there is no nonzero linear combination of the $\widetilde{h_i}(w_1, ..., w_t)$ being of the form $\sum a'_i w_i^2 + b'$, then we must have $s' \geq 10r$ and after the first elimination (i.e., via (17)), the expressions corresponding to $X_9^2 + X_{11}^2$ and $X_{10}^2 + X_{12}^2$ have been constants.

3.3 Finding the Plaintext

We substitute the linear expressions resulted by solving (21) into $\widetilde{h_1}(w_1, ..., w_t), \cdots, \widetilde{h_{15r}}(w_1, ..., w_t)$ and get $15r$ new quadratic polynomials on $12r - s' - p$ ($\leq 2r$) variables. Denote them by $\widehat{h_1}, \cdots, \widehat{h_{15r}}$. Since $12r - s' - p$ is very small (at most 8 and 10 for MFE-1 and MFE-1′, respectively), in principle, we can use the Gröbner basis method to solve the system

$$\widehat{h_i} = v'_i, \quad \forall\ i = 1, \cdots, 15r \tag{22}$$

very easily and to find the plaintext finally.

However, we know here that we start from $15r$ equations, therefore we expect that we will get much more than $2r$ (the number of variables) equations. This means we can solve it easily, for example, using the XL method [CKPS00]. In our experiments, this set of equations does turn out to be very easy to solve.

3.4 A Practical Attack Procedure, Its Complexity and Experimental Verification

Our attack can be divided into the following four steps:

Step 1 of the attack: *Find a basis of the linear space of the coefficient vectors $(a_{ijk}, b_{ij}, \cdots, f)$ of all SOLEs.*

As mentioned in §3.1, this is to solve a system of linear equations obtained by evaluating sufficiently many plain/cipher-texts in (9). There are $\binom{12r+1}{1}\binom{15r+2}{2}$ monomials of the form $u_i^\alpha v_j^\beta v_k^\gamma$ on u_i and v_j ($\alpha, \beta, \gamma = 0$ or 1). This number is 92659 and 178486 for $r = 4$ and 5, respectively, and is somewhat large. Choosing a number of plain/cipher-text pairs slightly more than the number of unknowns, say 1000, we can completely find the solution space in general. The complexity is respectively $92659^3 < 2^{50}$ and $178486^3 < 2^{52}$ using a naive Gaussian elimination.

This step is a one time computation for any given public key. Let $(a_{ijk}^{(l)}, b_{ij}^{(l)}, \cdots, f^{(l)})$, $1 \leq l \leq s$, be a basis of the equation system.

Our computer experiments confirm that that the dimension of SOLE is indeed $24r$ for both MFE-1 and MFE-1′.

Step 2 of the attack: *Given a valid ciphertext* $(v'_1, \cdots, v'_{15r})$*, we plug it into (17) and solve the system of linear equations to obtain linear expressions of the remaining* $12r - s'$ *in terms of the other* s' *variables of the plaintext components.*

The complexity of this step is $15rs^2 < (15r)^3$, and is less than 2^{19}.

Substitute these linear expressions into the original public key polynomials to get new quadratic polynomials $\widetilde{h_1}(w_1, ..., w_t), \cdots$, and $\widetilde{h_{15r}}(w_1, ..., w_t)$.

Step 3 of the attack: *Solve the system (20) and obtain its solution basis* $(\widetilde{a_1}^{(l)}, \cdots, \widetilde{a_{15r}}^{(l)}, \widetilde{b_1}^{(l)}, \cdots, \widetilde{b_t}^{(l)})$*,* $1 \leq l \leq p$*. Then solve the system (21) to find expression of the* p *components of the plaintext by the remaining* $12r - s' - p$ *linearly.*

The complexity of solving (20) is $(15r + t)^3 < (30r)^3 < 2^{22}$, and that for (21) is $pt^2 < (15r)^3 < 2^{19}$.

Our computer experiments show that indeed s' is 8r and p is $2r$ for both MFE-1 and MFE-1′.

Step 4 of the attack: *Derive new public key polynomials* $(\widehat{h_1}, \cdots, \widehat{h_{15r}})$ *from the solutions of (21), solve the system (22) and finally obtain the value of* p *components of the plaintext by using a Gröbner base or a linearization method. Then we use the linear expressions on the remaining plaintext components derived in in steps 2 and 3 to find the eliminated components.*

In 1000 experimental samples we have done, we find that after Step 3, the number of linearly independent quadratic equations are actually 20 for MFE-1 and 25 for MFE-1′. We actually solve them by finding a set of $2r$ linearly independent linear equations inside the space spanned by these equations. It takes no time.

Therefore the total attack complexity is less than 2^{52}. The complexity of the attack recovering the plaintext (steps 2, 3 and 4) is less than 2^{23}.

3.5 Extension of MFE and High Order Linearization Attack

The construction of MFE relies on the multiplicative structure of 2×2 matrices and it is not difficult to see that one can extend this construction in a straightforward way by using matrices of larger sizes $m \times m$, for example, 3×3 or 4×4, to build new MFE cryptosystems.

For any such an construction using matrix of $m \times m$, it is not difficult to see that the m-th order LE can be applied to attack the cryptosystem. The fundamental reason behind is the formula that for any matrix Q of size $m \times m$, we know that

$$Q^{-1} = \frac{1}{\mathrm{d}et(Q)} Q^*,$$

where Q^* is the associated matrix of Q.

In terms of algebraic formulas for $\mathrm{d}et(Q)$ and Q^*, we know that $\mathrm{d}et(Q)$ can be expressed as a degree m polynomial of the components Q_{ij} of Q and each component of Q^* can be expressed in terms of the a degree $m - 1$ polynomial

of the components Q_{ij} of Q. With this and the formulas (6) and (10) and other similar formulas, we can see that, for such a case, the order m linearization equations exists and they can be used to attack such a system. Therefore the current design of MFE needs to increase m substantially to avoid such an attack.

4 The Connection of HOLE with XL

One important point we want to make is that the HOLE method is closely related to the XL method [CKPS00]. In particular one may also explore the possibility of combining these two algebraic method together to develop additional techniques.

Assume we are given a system of equations

$$f_i(u_1, \cdots, u_n) = v'_i, \;\; 1 \le i \le m.$$

Let $U = (u_1, \cdots, u_n)$ and $g_i(U) = f_i(U) - v'_i$. For any nonnegative integral vector $\alpha = (\alpha_1, \cdots, \alpha_n)$, denote $u_1^{\alpha_1} \cdots u_n^{\alpha_n}$ by U^α. Similarly, for $\beta = (\beta_1, \cdots, \beta_m)$, denote $f_1^{\beta_1} \cdots f_m^{\beta_m}$ by F^β and $g_1^{\beta_1} \cdots g_m^{\beta_m}$ by G^β.

The XL method first translates the equation system above into another system of equations of the form

$$\sum a_{\alpha,i} U^\alpha g_i(U) = 0, \tag{23}$$

where $1 \le i \le m$ and α are nonnegative integral vectors with small component sum (upper-bounded by some small integer D). Then define all terms $U^\alpha U^\gamma$ as new unknowns and solve the resulted linear equation system.

On the other hand, the HOLE method attempts to solve a system of equations of the form

$$\sum_{i,\beta} a_{i,\beta} u_i G^\beta = 0, \tag{24}$$

where $1 \le i \le n$ and β are chosen small vectors. Since the $f_i(U)$ are equivalent to the $g_i(U)$ under affine transformations, the above system is equivalent to that of

$$\sum_{i,\beta} b_{i,\beta} u_i F^\beta = 0. \tag{25}$$

Our attack presented in the previous section actually finds **identical** equations with the form (25), and hence we can substitute F^β by $v_1'^{\beta_1} \cdots v_m'^{\beta_m}$ and get a linear system that the plaintext satisfies.

As a comparison, we find that if a HOLE with order D could be used to successfully attack a system by finding linear equations, then one should expect that the XL method should work as well. But the order of XL should be of degree $2D-1$ (the total degree is $2D+1$), because the v_i in general are of degree 2. From this consideration, we conclude that though HOLE definitely cannot be a replacement for the XL method, there could be cases that the HOLE method would be much more efficient than XL. In one case we consider polynomials of degree $D+1$ (HOLE), while in the other case, we consider polynomials of degree $2D+1$ (XL). Another critical point is that when we use the HOLE method, the

computation of HOLEs is performed only once for a given public key, then the HOLEs are used for any ciphertext; while the general XL algorithm needs to run its main part each time for different values of ciphertext. Thus one should think HOLE as a possibly more efficient alternative to XL, if it can work, and there should be cases that HOLE can work practically while the XL cannot.

More importantly, one may consider unifying the XL and HOLE methods. We may expect to efficiently solve the system of equations of the form:

$$\sum_{\alpha,\beta} a_{\alpha,\beta} U^{\alpha} G^{\beta} = 0. \tag{26}$$

From the point view of algebraic geometry, this definitely makes sense, which could allow us to produce more useful equations. We already have some idea about how this can be done, but at this moment, we have not yet found any example where such a method could indeed be more efficient in an attack on multivariate public key cryptosystems.

5 Conclusion

In this paper, we use an extension of the linearization equation attack method of Patarin, which we call the high order linearization equation method, to break the MFE multivariate public key cryptosystem in CT-RSA 2006. This shows that the high order linearization equation method is indeed an important algebraic attack method. For any multivariate public key cryptosystem, one should take into account this new method. Furthermore, one can expect that this method may be used to attack other cryptosystems, such as symmetric ciphers.

References

[ACDG03] Mehdi-Laurent Akkar, Nicolas T. Courtois, Romain Duteuil, and Louis Goubin. A fast and secure implementation of Sflash. In *PKC-2003, LNCS*, volume 2567, pages 267–278. Springer, 2003.

[AFIKS04] Gwénolé Ars and Jean-Charles Faugère and Hideki Imai and Mitsuru Kawazoe and Makoto Sugita Comparison between XL and Gröbner Basis Algorithms, *Asiacrypt 2004*, LNCS, V. 3329.

[MFSY05] M. Bardet, J-C. Faugre, B. Salvy and B-Y. Yang Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems MEGA 2005, Eighth International Symposium on Effective Methods in Algebraic Geometry, Porto Conte, Alghero, Sardinia (Italy), May 27th - June 1st, 2005, 15 pages.

[C00] Nicolas T. Courtois The security of hidden field equations (HFE) In C. Naccache, editor, Progress in cryptology, CT-RSA, LNCS, volume 2020, pages 266–281. Springer, 2001

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In B. Preenel, editor, *Advances in cryptology, Eurocrypt 2000, LNCS*, volume 1807, pages 392–407. Springer, 2000.

[CPi02] Nicolas Courtois, Josef Pieprzyk: Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. ASIACRYPT 2002, LNCS 2501, 267-287, Springer 2002.

[CM01] J. Chen and T. Moh. On the Goubin-Courtois attack on TTM. *Cryptology ePrint Archive*, 72, 2001. http://eprint.iacr.org/2001/072.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[Din04a] Jintai Ding. A new variant of the Matsumoto-Imai cryptosystem through perturbation. In F. Bao, R. Deng, and J. Zhou, editors, *the 7th International Workshop on Practice and Theory in Public key Cryptography, Singapore, (PKC'04), LNCS*, volume 2947, pages 305–318. Springer, 2004.

[DG05] Jintai Ding and Jason Gower Inoculating Multivariate Schemes Against Differential Attacks. Accepted for PKC-2006, IACR eprint 2005/255.

[DS03a] J. Ding and D. S. Schmidt. A common defect of the TTM cryptosystem. In *Proceedings of the technical track of the ACNS'03, ICISA Press*, pages 68–78, 2003. http://eprint.iacr.org.

[DS03b] J. Ding and D. S. Schmidt. The new TTM implementation is not secure. In H. Niederreiter K.Q. Feng and C.P. Xing, editors, *Proceedings of International Workshop on Coding, Cryptography and Combinatorics (CCC 2003)*, pages 106–121, 2003.

[DS04a] Jintai Ding, and D. S. Schmidt Cryptanalysis of HFEV and the internal perturbation of HFE. The 8th International Workshop on Practice and Theory in Public key Cryptography, Jan. 2005, Switzerland (PKC'05), Lecture Notes in Computer Sciences, volume 3386, pages 288-301 Springer, 2005.

[DS05] Jintai Ding, and D. S. Schmidt Rainbow, a new multivariate public key signature scheme. The Third International Conference of Applied Cryptography and Network Security (ACNS 2005), New York, June 7-10, 2005, Lecture Notes in Computer Science 3531, Page 164-175, Springer, 2005.

[Fau99] Jean-Charles Faugère A new efficient algorithm for computing Gröbner bases (F_4), Journal of Pure and Applied Algebra, V. 139, P. 61-88, June 199.

[FGS05] P.-A. Fouque, L. Granboulan, and J. Stern. Differential Cryptanalysis for Multivariate Schemes Advances in Cryptology - EUROCRYPT 2005, Lecture Notes in Computer Science 3494 Springer 2005, Page 341-353.

[GC00] L. Goubin and N. Courtois. Cryptanalysis of the TTM cryptosystem. *LNCS, Springer Verlag*, 1976:44–57, 2000.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In M. Wiener, editor, *Advances in crytology – Crypto '99, LNCS*, volume 1666, pages 19–30. Springer, 1999.

[MI88] T. Matsumoto and H. Imai. Public quadratic polynomial-tuples for efficient signature verification and message encryption. In C. G. Guenther, editor, *Advances in cryptology – EUROCRYPT'88, LNCS*, volume 330, pages 419–453. Springer, 1988.

[Moh99] T. T. Moh. A fast public key system with signature and master key functions. *Lecture Notes at EE department of Stanford University.*, May 1999. http://www.usdsi.com/ttm.html.

[MCY04] T.Moh and J.M.Chen and Boyin Yang Building Instances of TTM Immune to the Goubin-Courtois Attack and the Ding-Schmidt Attack. IACR eprint 2004/168, http://eprint.iacr.org.

[NES] NESSIE. European project IST-1999-12324 on New European Schemes for Signature, Integrity and Encryption. http://www.cryptonessie.org.

[Pat95] J. Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88. In D. Coppersmith, editor, *Advances in Cryptology – Crypto '95, LNCS*, volume 963, pages 248–261, 1995.

[Pat96] J. Patarin. Hidden field equations (HFE) and isomorphism of polynomials (IP): Two new families of asymmetric algorithms. In U. Maurer, editor, *Eurocrypt'96, LNCS*, volume 1070, pages 33–48. Springer, 1996.

[Pat97] J. Patarin. The oil and vinegar signature scheme. *Dagstuhl Workshop on Cryptography, September 1997*, 1997.

[PCG01a] Jacques Patarin, Nicolas Courtois, and Louis Goubin. Flash, a fast multivariate signature algorithm. In *LNCS*, volume 2020, pages 298–307. Springer, 2001.

[PGC98] Jacques Patarin, Louis Goubin, and Nicolas Courtois. C^*_{-+} and HM: variations around two schemes of T. Matsumoto and H. Imai. In K. Ohta and D. Pei, editors, *ASIACRYPT'98, LNCS*, volume 1514, pages 35–50. Springer, 1998.

[PQ] PQCrypto 2006: International Workshop on Post-Quantum Cryptography http://postquantum.cr.yp.to/

[RSA78] Ronald Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public key cryptosystems. *ACM*, 21(2):120–126, 1978.

[Sho99] Peter Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.*, 41(2):303–332, 1999.

[WHLCY05] Lih-Chung Wang and Yuh-Hua Hu and Feipei Lai and Chun-Yen Chou and Bo-Yin Yang, Tractable Rational Map Signature, Public Key Cryptosystems 2005, LNCS 3386, Springer, P. 244-257.

[WYH06] Lih-Chung Wang, Bo-yin Yang, Yuh-Hua Hu and Feipei Lai, A Medium-Field Multivariate Public key Encryption Scheme, CT-RSA 2006: The Cryptographers' Track at the RSA Conference 2006, LNCS 3860, 132-149, Springer, 2006.

[YC05] B. Yang and J. Chen. Building Secure Tame-like Multivariate Public key Cryptosystems–The New TTS. Information Security and Privacy: 10th Australasian Conference–ACISP 2005, LNCS 3574, 2005, Springer, P. 518-531.

Proposal for Piece In Hand Matrix Ver.2: General Concept for Enhancing Security of Multivariate Public Key Cryptosystems

Shigeo Tsujii[†] Kohtaro Tadaki[‡] Ryou Fujita[†]

[†] Institute of Information Security
2–14–1 Tsuruya-cho, Kanagawa-ku, Yokohama-shi, 221–0835 Japan
[‡] 21st Century Center Of Excellence Program, Chuo University
1–13–27 Kasuga, Bunkyo-ku, Tokyo, 112–8551 Japan

Abstract. We proposed the concept, *piece in hand (soldiers in hand) matrix* and have developed the framework based on the concept so far. The piece in hand (PH, for short) matrix is a general concept which can be applicable to any type of multivariate public key cryptosystems to enhance their security. In this paper, we make improvements in the PH matrix method as follows. (i) In the PH matrix method, an arbitrary number of additional variables can be introduced to the random polynomial term in the public key, which is eliminated by the multiplication of the public key by the PH matrix during the decryption. Thus these additional variables enables the public key to have more than one solution, and therefore increases the difficulty to solve the public key. We show, in an experimental manner, that the PH matrix method improved in this way is secure even against the Gröbner basis attack. (ii) In the nonlinear PH matrix method proposed previously, the degree of polynomials in the public key is more than two, and this may cause an undesirable increase in the size of the public key. In this paper, we propose a nonlinear PH matrix method, where the degree of polynomials in the public key is kept the same as the degree of polynomials in the public key of the original cryptosystem, which is normally two.

Key words: public key cryptosystem, multivariate polynomial, multivariate public key cryptosystem, piece in hand concept, soldiers in hand

1 Introduction

The research of multivariate public key cryptosystems started with the works by Matsumoto et al. [13] in 1983 and Tsujii et al. [18] in 1986.[1] Especially, the multivariate public key cryptosystem which was proposed by Matsumoto and Imai [14] in 1988 is known as *the Matsumoto-Imai cryptosystem* nowadays. In 1995 Patarin constructed an attack against the Matsumoto-Imai cryptosystem in a heuristic manner [16] and then proposed an improvement of the cryptosystem, called *HFE*, in 1996 [17]. Subsequently, Kipnis and Shamir introduced a general technique, called *relinearization*, to solve system of multivariate polynomial equations, and used it to attack HFE in 1999

[1] This paper deals only with encryption schemes and not signature schemes. The piece in hand matrix method, which is considered in this paper, can be applied to signature schemes in general as well. For simplicity, however, we only describe the application of the method to encryption schemes in this paper.

[12]. Recently, Faugère and Joux showed in an experimental manner that computing a Gröbner basis of the public key is likely to be an efficient attack to HFE [5]. Because of the simplicity of this attack, it may be a threat to many types of proposed multivariate public key cryptosystems, and now seems to be thought of as one of the major attacks against multivariate public key cryptosystems.

On the other hand, in the work [18] Tsujii et al. proposed a multivariate public key cryptosystem by introducing a trapdoor called *the sequential solution method*. The cryptosystem was then broken by Hasegawa and Kaneko [7] for the special case where rational functions are used. Later on, in 1989, [19] proposed the revised version of [18], where birational transformation, named *core transformation*, was employed.[2] No attack to this revised version has been succeeded so far.

In 2000 Kasahara and Sakai proposed a multivariate public key cryptosystem with random variables [8] and have improved the cryptosystem in a series of works such as [9, 10, 11]. Wolf, Braeken, and Preneel [25] introduced the notion of a class of multivariate public key cryptosystems called *the stepwise triangular systems* which is a generalization of the cryptosystems RSE(2)PKC [9] and RSSE(2)PKC [10] proposed by Kasahara and Sakai. They then showed, in the same paper [25], that the stepwise triangular systems, which includes RSE(2)PKC and RSSE(2)PKC, to be by no means secure. On the other hand, by introducing random terms to the Matsumoto-Imai cryptosystem, Ding proposed a multivariate public key cryptosystem called *the perturbed Matsumoto-Imai cryptosystem* [4]. (See e.g. [26] for the bibliographical details of multivariate public key cryptosystems in general.)

In 2003 one of us [20] proposed the concept, *piece in hand matrix*. Thereafter we have developed the framework based on the concept in a series of works [21, 22, 23, 24] so far. The concept of the piece in hand (PH, for short) matrix has the following properties: (i) The PH matrix is a general concept which can be applicable to any type of multivariate public key cryptosystems to enhance their security. (ii) In a framework of the PH matrix, the original public key, which is represented by a polynomial vector, is randomized by adding a random polynomial term and then published. In the decryption, the legitimate receiver can obtain the cipher text of the original cryptosystem by multiplying the PH matrix and eliminating the random term, and then can recover the plain text according to the decryption of the original cryptosystem. (iii) There are two types of the PH matrices: a linear matrix whose elements are constants and a nonlinear matrix whose elements are functions of the plain text or random numbers.

In this paper, we make improvements in the PH matrix methods as follows: (i) In PH matrix methods in general, an arbitrary number of additional variables can be introduced to the random polynomial term in the public key, which is eliminated by the multiplication of the public key by the PH matrix during the decryption. Thus the number of variables can be increased more than the number of polynomials in the public key, where random numbers are substituted to the additional variables on the encryption, and therefore these additional variables in the public key can be made to have exponentially many solutions. We show in an experimental manner that the PH matrix method improved in this way is secure even against the Gröbner basis attack. (ii) In the nonlinear PH matrix method [22] proposed previously, the degree of polynomials in the public key is more than two, and this may cause an undesirable increase in the size of the public key. In this paper, we propose a nonlinear PH matrix method, where the degree of polynomials in the public key is kept the same as the degree of polynomials in the public key of the original cryptosystem, which is

[2] The paper [19] was originally written in Japanese. An English translation of [19] is included in [22] as an appendix.

normally two.

1.1 Organization

This paper is organized as follows. We begin in Section 2 with some basic notation and a brief introduction of the schemes of multivariate public key cryptosystems in general. In Section 3, we recall the primitive form of the general prescription for enhancing the security of any given multivariate public key cryptosystem by the linear PH matrix method, introduced in [22]. We then reconsider the enhancement of security against the Gröbner basis attack. Based on the consideration, as one of the countermeasures against the Gröbner basis attack, we propose a new linear PH matrix method with random variables in Section 4. We then show, based on computer experiments, that the new method properly provides substantial robustness against the Gröbner basis attack. In Section 5, we present another countermeasure against the Gröbner basis attack by improving the nonlinear PH matrix method originally proposed in the work [22]. This new nonlinear PH matrix method is a practical one, compared with the previous proposal, since it holds down an undesirable increase in the size of the public key. We conclude this paper with a discussion about the future direction of our work in Section 6.

2 Preliminaries

In this preliminary section we review the schemes of multivariate public key cryptosystems in general after introducing some notation about fields, polynomials, and matrices.

2.1 Notation

$\mathbf{F}_q$ is a finite field which has q elements with $q \geq 2$. $\mathbf{F}_q[x_1, \ldots, x_k]$ is the set of all polynomials in variables $x_1, x_2, \ldots, x_k$ with coefficients in $\mathbf{F}_q$. For every nonempty set S and every positive integers n and l, $S^{n \times l}$ denotes the set of all $n \times l$ matrices whose entries are in S, and S^n denotes the set of all column vectors consisting n components in S. Therefore $S^{n \times 1} = S^n$. We represent a column vector in general by bold face symbols such as $\boldsymbol{p}$, $\boldsymbol{E}$, and $\boldsymbol{X}$. For every matrix $A \in S^{n \times l}$, $A^T \in S^{l \times n}$ denotes the transpose of A. Let

$$\boldsymbol{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}$$

and

$$\boldsymbol{g} = \begin{pmatrix} g_1 \\ \vdots \\ g_k \end{pmatrix}$$

be polynomial column vectors in $\mathbf{F}_q[x_1, \ldots, x_k]^n$ and $\mathbf{F}_q[x_1, \ldots, x_m]^k$, respectively, where $f_1, \ldots, f_n \in \mathbf{F}_q[x_1, \ldots, x_k]$ and $g_1, \ldots, g_k \in \mathbf{F}_q[x_1, \ldots, x_m]$. Then the *substitution* $\boldsymbol{f}(\boldsymbol{g}) \in \mathbf{F}_q[x_1, \ldots, x_m]^n$ of $\boldsymbol{g}$

for the variables in $\boldsymbol{f}$ is defined by

$$\boldsymbol{f}(\boldsymbol{g}) \equiv \begin{pmatrix} h_1 \\ \vdots \\ h_n \end{pmatrix},$$

where each h_i is the polynomial in $\mathbf{F}_q[x_1, \ldots, x_m]$ obtained by substituting $g_1, \ldots, g_k$ for the variables $x_1, \ldots, x_k$ in f_i, respectively. Thus, for every $\boldsymbol{f} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ and every $\boldsymbol{p} \in {\mathbf{F}_q}^k$, $\boldsymbol{f}(\boldsymbol{p})$ denotes simply the vector in ${\mathbf{F}_q}^n$ obtained by substituting $p_1, \ldots, p_k$ for the variables $x_1, \ldots, x_k$ in $\boldsymbol{f}$, respectively, where $\boldsymbol{p} = (p_1, \ldots, p_k)^T$ with $p_1, \ldots, p_k \in \mathbf{F}_q$. For every polynomial matrix $N \in \mathbf{F}_q[x_1, \ldots, x_k]^{n \times l}$ and every polynomial column vector $\boldsymbol{g} \in \mathbf{F}_q[x_1, \ldots, x_m]^k$, the substitution $N(\boldsymbol{g}) \in \mathbf{F}_q[x_1, \ldots, x_m]^{n \times l}$ of $\boldsymbol{g}$ for the variables in N is defined in the same manner.

2.2 Schemes of Multivariate Public Key Cryptosystems

A multivariate public key cryptosystem such as in [13, 18, 14, 19, 17, 15, 8, 4, 9, 10, 25, 11] can be considered to comply with the following scheme: A plain text is represented by a column vector $\boldsymbol{p} = (p_1, \ldots, p_k)^T \in {\mathbf{F}_q}^k$, and a cipher text is represented by a column vector $\boldsymbol{c} = (c_1, \ldots, c_n)^T \in {\mathbf{F}_q}^n$. Then, q, k, and a polynomial column vector $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ form the public key of the cryptosystem. The encryption is given by the following transformation from $\boldsymbol{p}$ to $\boldsymbol{c}$:

$$\boldsymbol{c} = \boldsymbol{E}(\boldsymbol{p}).$$

The secret key of the cryptosystem gives an efficient method to solve the system $\boldsymbol{E} = \boldsymbol{c}$ of polynomial equations on $(x_1, \ldots, x_k)$ for any given $\boldsymbol{c} \in {\mathbf{F}_q}^n$. Thus, $\boldsymbol{E}$ has to be constructed so that, without the knowledge about this method, it is difficult to find $\boldsymbol{p}$ for any given $\boldsymbol{c}$ in polynomial-time.

Let us consider the situation that the legitimate receiver, Bob, has the secret key and the sender, Alice, wants to transmit her cipher text $\boldsymbol{c} \equiv \boldsymbol{E}(\boldsymbol{p})$ to Bob. When Bob receives the cipher text $\boldsymbol{c}$ sent from Alice, using the secret key he can efficiently decipher it to obtain the plain text $\boldsymbol{p}$. On the other hand, it has to be intractable for the eavesdropper, Catherine, to recover $\boldsymbol{p}$ from $\boldsymbol{c}$, based on the fact that she has no knowledge about the secret key.

In most multivariate public key cryptosystems, the public key $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ has the following form:

$$\boldsymbol{E} = B\,\boldsymbol{G}(A\boldsymbol{x}). \tag{1}$$

Here $\boldsymbol{x}$ denotes $(x_1, \ldots, x_k)^T \in \mathbf{F}_q[x_1, \ldots, x_k]^k$. A and B are invertible matrices in ${\mathbf{F}_q}^{k \times k}$ and ${\mathbf{F}_q}^{n \times n}$, respectively. $\boldsymbol{G}$ is a polynomial column vector in $\mathbf{F}_q[x_1, \ldots, x_k]^n$, where the polynomial vector $A\boldsymbol{x} \in \mathbf{F}_q[x_1, \ldots, x_k]^k$ is substituted for the variables $x_1, \ldots, x_k$ in $\boldsymbol{G}$ according to our convention above. In this type of cryptosystem, while keeping A and B secret (and so $\boldsymbol{G}$ in some cryptosystems) from anyone else, Bob publishes the public key $\boldsymbol{E}$ in the form of a system of trimmed multivariate polynomials obtained by simplifying the right-hand side of (1). Normally, $\boldsymbol{G}$ consists only of polynomials in $\mathbf{F}_q[x_1, \ldots, x_k]$ of total degree at most two in order to avoid the blowup of the size of the public key $\boldsymbol{E}$. In such a case, the multivariate public key cryptosystem is called a *quadratic multivariate public key cryptosystem.*

3 The Previous Method and the Gröbner Basis Attack

In this section, we recall the primitive form of the general prescription for enhancing the security of any given multivariate public key cryptosystem by our linear PH matrix method, introduced by [22]. We then reconsider the enhancement of security against the Gröbner basis attack and explore possible improvements in the linear PH matrix method.

3.1 General Prescription for Enhancement by the linear PH Matrix Method

The primitive form of the general prescription for the enhancement by the linear PH matrix method [22] is described as follows.

Let $\mathcal{K}$ be an arbitrary quadratic multivariate public key cryptosystem whose public key is given by $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$, as described in Subsection 2.2. In our linear PH matrix method, we construct a new quadratic multivariate public key cryptosystem $\widetilde{\mathcal{K}}$ from the given multivariate public key cryptosystem $\mathcal{K}$ for the purpose of enhancing the security. A public key $\widetilde{\boldsymbol{E}} \in \mathbf{F}_q[x_1, \ldots, x_k]^l$ of $\widetilde{\mathcal{K}}$ is constructed from the original public key $\boldsymbol{E}$ of $\mathcal{K}$ by the transformation:

$$\widetilde{\boldsymbol{E}} \equiv S\boldsymbol{E} + R\boldsymbol{X}. \tag{2}$$

Here $\boldsymbol{X}$ denotes the column vector whose components are all monomials in $\mathbf{F}_q[x_1, \ldots, x_k]$ of total degree at most two, enumerated in any order. Thus, $\boldsymbol{X}$ can be chosen as

$$\boldsymbol{X} \equiv (x_1x_1, x_1x_2, \ldots, x_{k-1}x_k, x_kx_k, x_1, x_2, \ldots, x_k, 1)^T.$$

S is a matrix in ${\mathbf{F}_q}^{l\times n}$. In order to make our PH matrix method work properly, we assume that $l > n$. On the other hand, R is a matrix in ${\mathbf{F}_q}^{l\times t}$, where t is the number of components of $\boldsymbol{X}$. Note that $t = \binom{k+2}{2} = (k^2 + 3k + 2)/2$ for $q \geq 3$.[3] The term $R\boldsymbol{X}$ plays a role in randomizing $\widetilde{\boldsymbol{E}}$. Hence R has to be chosen so that neither the hidden original public key $\boldsymbol{E}$ nor any equivalent polynomial vector which enables an eavesdropper to obtain Alice's plain text can be identified in the actual public key $\widetilde{\boldsymbol{E}}$. A plain text of $\widetilde{\mathcal{K}}$ is represented by a vector in ${\mathbf{F}_q}^k$ in the same way as in $\mathcal{K}$. For any plain text vector $\boldsymbol{p} \in {\mathbf{F}_q}^k$ of $\widetilde{\mathcal{K}}$, the corresponding cipher text of $\widetilde{\mathcal{K}}$ is represented by a vector $\widetilde{\boldsymbol{c}} \in {\mathbf{F}_q}^l$ and is calculated by $\widetilde{\boldsymbol{c}} = \widetilde{\boldsymbol{E}}(\boldsymbol{p})$.

In addition to the matrices S and R, we introduce the *PH matrix* $M \in {\mathbf{F}_q}^{n\times l}$ as a secret key of $\widetilde{\mathcal{K}}$. In the key-generation stage, the matrices R, M, and S are chosen in sequence so as to satisfy the following three conditions. We can show that this choice is efficiently possible.

Condition 1. $l \geq n + \operatorname{rank} R$. □

Condition 2. $MR = 0$ *and* $\operatorname{rank} M = n$. □

Condition 3. $MS = I_n$, *where* I_n *is the identity matrix in* ${\mathbf{F}_q}^{n\times n}$. □

By the above conditions we see that the multiplication of $\widetilde{\boldsymbol{E}}$ by the PH matrix M from the left simplifies $\widetilde{\boldsymbol{E}}$ and results in the original public key $\boldsymbol{E}$ as follows:

$$M\widetilde{\boldsymbol{E}} = \boldsymbol{E}. \tag{3}$$

[3] In the case of $q = 2$, we can eliminate the monomials $x_1^2, \ldots, x_k^2$ from $\boldsymbol{X}$ using the so-called *field equations* $x_i^2 = x_i$ $(i = 1, \ldots, k)$. Thus t is calculated to be $(k^2 + k + 2)/2$ in such a case.

This is a crucial property of the PH matrix in our PH matrix methods in general.

Then, the triple $(q, k, \widetilde{\boldsymbol{E}})$ forms the public key of $\widetilde{\mathcal{K}}$. On the other hand, the PH matrix M, together with the secret key of $\mathcal{K}$ corresponding to the public key $(q, k, \boldsymbol{E})$ of $\mathcal{K}$, forms the secret key of $\widetilde{\mathcal{K}}$. The decryption of $\widetilde{\mathcal{K}}$ proceeds as follows. Based on the relation (3), on receiving the cipher text $\widetilde{\boldsymbol{c}} \equiv \widetilde{\boldsymbol{E}}(\boldsymbol{p})$ for a plain text $\boldsymbol{p}$, Bob can efficiently calculate $\boldsymbol{c} \equiv \boldsymbol{E}(\boldsymbol{p}) = M\widetilde{\boldsymbol{c}}$ by the multiplication of $\widetilde{\boldsymbol{c}}$ by M from the left. Then, according to the decryption procedure of $\mathcal{K}$, Bob can recover the plain text $\boldsymbol{p}$ using the secret key of $\mathcal{K}$.

The encryption and decryption processes in the PH matrix method are schematically represented in Figure 1.

- Encryption

$$\underset{l\times 1 \text{ vector}}{\boxed{\widetilde{\boldsymbol{E}}}} = \underset{l\times n \text{ matrix}}{\boxed{S}} \underset{n\times 1 \text{ vector}}{\boxed{\boldsymbol{E}}} + \underset{l\times t \text{ matrix}}{\boxed{R}} \underset{t\times 1 \text{ vector}}{\boxed{X}}$$

- Decryption

$$\underset{n\times l \text{ matrix}}{\boxed{\begin{matrix}\text{PH matrix}\\ M\end{matrix}}} \boxed{\widetilde{\boldsymbol{E}}} = \boxed{\boldsymbol{E}} \qquad \begin{cases} MS = I_n, \\ MR = 0. \end{cases}$$

Figure 1: The encryption and decryption in the PH matrix method

3.2 Countermeasures against the Gröbner Basis Attack

Recently, Faugère and Joux [5] showed in an experimental manner that computing a Gröbner basis of the public key is likely to be an efficient attack to HFE [17], which is one of the major variants of multivariate public key cryptosystem. The attack is simply to compute a Gröbner basis for the ideal generated by polynomial components in $\boldsymbol{E} - \boldsymbol{c}$, where $\boldsymbol{E}$ is a public key and $\boldsymbol{c}$ is a cipher text vector. Thus, because of the simplicity of this attack, it may be a threat to many types of proposed multivariate public key cryptosystems.

Especially, from the point of view of Gröbner bases, the secret invertible matrix B may be useless in a scheme whose public key has the form (1). This is because any ideal I generated by polynomials remains unchanged under the transformation of the generators of I by an invertible matrix. Thus, by the following reason, the PH matrix concept might be also useless to the Gröbner basis attack in its primitive implementation presented in the previous subsection. We first note, by the relation (3), that $M(\widetilde{\boldsymbol{E}} - \widetilde{\boldsymbol{c}}) = \boldsymbol{E} - \boldsymbol{c}$, where $\widetilde{\boldsymbol{c}} \equiv \widetilde{\boldsymbol{E}}(\boldsymbol{p}) \in {\mathbf{F}_q}^l$ is a cipher text vector of

the enhanced cryptosystem $\widetilde{\mathcal{K}}$ and $\boldsymbol{c} \equiv \boldsymbol{E}(\boldsymbol{p}) \in {\mathbf{F}_q}^n$ is the corresponding cipher text vector of the original cryptosystem $\mathcal{K}$. We can then show that there exist linear combinations $g_1, \ldots, g_{l-n}$, with coefficients in $\mathbf{F}_q$, of $\widetilde{e}_1 - \widetilde{c}_1, \ldots, \widetilde{e}_l - \widetilde{c}_l$ such that

$$\langle \widetilde{e}_1 - \widetilde{c}_1, \ldots, \widetilde{e}_l - \widetilde{c}_l \rangle = \langle e_1 - c_1, \ldots, e_n - c_n, g_1, \ldots, g_{l-n} \rangle, \tag{4}$$

where $(c_1, \ldots, c_n)^T \equiv \boldsymbol{c}$ and $(\widetilde{c}_1, \ldots, \widetilde{c}_l)^T \equiv \widetilde{\boldsymbol{c}}$, and $(e_1, \ldots, e_n)^T \equiv \boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ and $(\widetilde{e}_1, \ldots, \widetilde{e}_l)^T \equiv \widetilde{\boldsymbol{E}} \in \mathbf{F}_q[x_1, \ldots, x_k]^l$ are the public keys of $\mathcal{K}$ and $\widetilde{\mathcal{K}}$, respectively. Thus, from the point of view of Gröbner bases, the system $\widetilde{\boldsymbol{E}} - \widetilde{\boldsymbol{c}} = \boldsymbol{0}$ of polynomial equations on $(x_1, \ldots, x_k)$ might not be necessarily more difficult to solve than the system $\boldsymbol{E} - \boldsymbol{c} = \boldsymbol{0}$ of polynomial equations on $(x_1, \ldots, x_k)$ due to the existence of the additional polynomial equations $g_1 = 0, \ldots, g_{l-n} = 0$ for the former. In such a case, the linear PH matrix method might be useless to the Gröbner basis attack. This paper proposes two types of new PH matrix methods which may overcome this weakness, through elaborations of the original linear PH matrix method described in the previous subsection.

In the next section, we introduce a linear PH matrix method with random variables and consider its security. In the above consideration, the polynomials $e_1, \ldots, e_n$ are assumed to be in $\mathbf{F}_q[x_1, \ldots, x_k]$ implicitly, and therefore the weakness of the original linear PH matrix method against the Gröbner basis attack is of concern. Hence, one of the countermeasures against the weakness is to introduce additional variables $x_{k+1}, \ldots, x_m$ to the public key $\widetilde{\boldsymbol{E}}$ of $\widetilde{\mathcal{K}}$. Under this countermeasure, the g_i's in (4) are no longer polynomials in $\mathbf{F}_q[x_1, \ldots, x_k]$, but in $\mathbf{F}_q[x_1, \ldots, x_m]$, and therefore solving the system $\widetilde{\boldsymbol{E}} - \widetilde{\boldsymbol{c}} = \boldsymbol{0}$ of polynomial equations on $(x_1, \ldots, x_m)$ seems to be more difficult than solving the system $\boldsymbol{E} - \boldsymbol{c} = \boldsymbol{0}$ of polynomial equations on $(x_1, \ldots, x_k)$. This is done by introducing to the term $R\boldsymbol{X}$ in (2) the additional variables $x_{k+1}, \ldots, x_m$ which are set to random values by Alice on the encryption. In the next section, we describe a new linear PH matrix method based on this idea, and we show that the new method properly works and provides substantial robustness against the Gröbner basis attack, based on computer experiments.

In Section 5 we also present another countermeasure against the Gröbner basis attack through a nonlinearization of the PH matrix. In the previous work [22], we already proposed a nonlinear PH matrix method. However, the degree of polynomials in the public key of the enhanced cryptosystem $\widetilde{\mathcal{K}}$ is more than two in the previous method, and this may cause an undesirable increase in the size of the public key of the enhanced cryptosystem. In a new nonlinear PH matrix method, the degree of polynomials in the public key of $\widetilde{\mathcal{K}}$ is always the same as the degree of polynomials in the public key of the original cryptosystem $\mathcal{K}$. Thus, the new nonlinear PH matrix method is a practical one, compared with the previous proposal.

4 Linear PH Matrix Method with Random Variables

In this section, as a countermeasure against the Gröbner basis attack, we introduce a new linear PH matrix method with random variables, based on the primitive linear PH matrix method described in Subsection 3.1. The point of the modification is to introduce additional variables to the public key of the enhanced cryptosystem. By this countermeasure, the computational complexity of the Gröbner basis attack is likely to increase exponentially in the number of the additional variables, as is suggested by the experimental results reported below.

4.1 The New Linear PH Matrix Method

Let $\mathcal{K}$ be an arbitrary quadratic multivariate public key cryptosystem whose public key is given by $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$, as described in Subsection 2.2. We construct a new quadratic multivariate public key cryptosystem $\widetilde{\mathcal{K}}$ based on $\mathcal{K}$ as follows for the purpose of enhancing the security even against the Gröbner basis attack. Let p and m be any positive integers with $p < k < m$.

Key-Generation. In the key-generation stage, the public key and secret key of $\mathcal{K}$ are chosen first. Then, a public key $\widetilde{\boldsymbol{E}} \in \mathbf{F}_q[x_1, \ldots, x_m]^l$ of $\widetilde{\mathcal{K}}$ is constructed from the original public key $\boldsymbol{E}$ of $\mathcal{K}$ by the following transformation:

$$\widetilde{\boldsymbol{E}} \equiv S\boldsymbol{E}\begin{pmatrix} \boldsymbol{x} \\ A\boldsymbol{z} \end{pmatrix} + R\boldsymbol{Z}, \tag{5}$$

where $\boldsymbol{x} = (x_1, \ldots, x_p)^T \in \mathbf{F}_q[x_1, \ldots, x_p]^p$ and $\boldsymbol{z} = (x_1, \ldots, x_m)^T \in \mathbf{F}_q[x_1, \ldots, x_m]^m$. A is a randomly chosen matrix in ${\mathbf{F}_q}^{(k-p)\times m}$. Note that, in the right-hand side of (5), the vector $A\boldsymbol{z} \in \mathbf{F}_q[x_1, \ldots, x_m]^{k-p}$ is substituted for the variables $x_{p+1}, \ldots, x_k$ in the original public key $\boldsymbol{E}$ while keeping the variables $x_1, \ldots, x_p$ in $\boldsymbol{E}$ unchanged. This $A\boldsymbol{z}$ is needed to prevent an eavesdropper from forging the PH matrix, as is explained in Subsection 4.3 below. $\boldsymbol{Z}$ denotes the column vector whose components are all monomials in $\mathbf{F}_q[x_1, \ldots, x_m]$ of total degree at most two, enumerated in any order. Thus, $\boldsymbol{Z}$ can be chosen as

$$\boldsymbol{Z} \equiv (x_1x_1, x_1x_2, \ldots, x_{m-1}x_m, x_mx_m, x_1, x_2, \ldots, x_m, 1)^T.$$

S is a matrix in ${\mathbf{F}_q}^{l\times n}$. In order to make this PH matrix method work properly, we assume that $l > n$. On the other hand, R is a matrix in ${\mathbf{F}_q}^{l\times s}$, where s is the number of components of $\boldsymbol{Z}$. Note that $s = (m^2 + 3m + 2)/2$ for $q \geq 3$. In addition to the matrices S and R, we introduce the PH matrix $M \in {\mathbf{F}_q}^{n\times l}$ as a secret key of $\widetilde{\mathcal{K}}$. The matrices R, M, and S are randomly chosen in sequence so as to satisfy Conditions 1, 2, and 3 in Subsection 3.1. Note that, as in the case of the original method, this choice can be efficiently possible.

Then, the quadruple $(q, m, \widetilde{\boldsymbol{E}}, p)$ forms the public key of $\widetilde{\mathcal{K}}$. Bob publishes it after the key-generation. On the other hand, the PH matrix M, together with the secret key of $\mathcal{K}$ corresponding to the public key $(q, k, \boldsymbol{E})$ of $\mathcal{K}$, forms the secret key of $\widetilde{\mathcal{K}}$.

Encryption. A plain text of $\widetilde{\mathcal{K}}$ is represented by a vector in ${\mathbf{F}_q}^p$. Now, assume that Alice wants to send Bob a plain text vector $\boldsymbol{p} \in {\mathbf{F}_q}^p$. The corresponding cipher text of $\widetilde{\mathcal{K}}$ is represented by a vector $\widetilde{\boldsymbol{c}} \in {\mathbf{F}_q}^l$, and is calculated by Alice through

$$\widetilde{\boldsymbol{c}} \equiv \widetilde{\boldsymbol{E}}\begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{r} \end{pmatrix}, \tag{6}$$

where $\boldsymbol{r} \in {\mathbf{F}_q}^{m-p}$ is chosen randomly by Alice on the encryption of $\boldsymbol{p}$.

Decryption. The decryption of $\widetilde{\mathcal{K}}$ proceeds as follows. We first note that, by Conditions 2 and 3,

$$M\widetilde{\boldsymbol{E}} = \boldsymbol{E}\begin{pmatrix} \boldsymbol{x} \\ A\boldsymbol{z} \end{pmatrix}, \tag{7}$$

where $\boldsymbol{x} = (x_1, \ldots, x_p)^T \in \mathbf{F}_q[x_1, \ldots, x_p]^p$ and $\boldsymbol{z} = (x_1, \ldots, x_m)^T \in \mathbf{F}_q[x_1, \ldots, x_m]^m$. Thus, on receiving the cipher text $\widetilde{\boldsymbol{c}}$ corresponding to a plain text $\boldsymbol{p}$, Bob can efficiently obtain the value

$$\boldsymbol{E}\begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{d} \end{pmatrix}$$

from the multiplication of $\widetilde{\boldsymbol{c}}$ by M, where $\boldsymbol{d}$ is a column vector in ${\mathbf{F}_q}^{k-p}$ given by

$$\boldsymbol{d} \equiv A\begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{r} \end{pmatrix}.$$

Then, according to the decryption procedure of $\mathcal{K}$, Bob can efficiently recover the plain text $\boldsymbol{p}$ using the secret key of $\mathcal{K}$. Note that $\boldsymbol{d}$ is discarded after the decryption.

4.2 Strength against the Gröbner Basis Attack

Based on the experimental results, we show that the linear PH matrix method described in the previous subsection may be secure against the Gröbner basis attack.

For any cipher text vector $\widetilde{\boldsymbol{c}}$, the corresponding plain text vector $\boldsymbol{p}$ is unique in (6). On the other hand, $\boldsymbol{r}$ is not necessarily unique, since A is not necessarily invertible and R is chosen randomly. The nonuniqueness of $\boldsymbol{r}$ may provides substantial robustness against the Gröbner attack, as is suggested by the experimental results shown below.

cryptosystems	p	k	m	l	running-times in second
HFE		10			< 1
$(128 < d < 513)$		25			686
		28			1404
the enhanced HFE	10	20	30	25	1364
by the PH method	10	20	35	25	5301
$(d < 513)$	10	20	37	25	8788
	10	20	32	28	3437
	10	20	36	28	9903
	10	20	38	28	15091

Table 1: Comparison between running-times for HFE and the enhanced HFE by the PH method.

We report in Table 1 the time required to compute reduced Gröbner bases of the public keys both of HFE and of the HFE enhanced by the linear PH method with random variables. The running-times are given for hp AlphaServer ES45 workstation with Alpha 21264 (EV68) processor at 1250 MHz and 32GB of RAM. We use the algorithm F_4 implemented on the computational algebra system Magma V2.12-14. Note that $n = k$ and $q = 2$ for the public keys $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ of HFE by its specifications. In the table, d denotes the degree of the univariate polynomial in the encryption of the HFE scheme. In the lower half of the table, the linear PH matrix method with random variables is applied to the public keys of HFE with $q = 2$, $n = k = 20$, and $\operatorname{rank} R = l - n$. The table shows that the increase of the number $m-k$ of random variables $x_{k+1}, \ldots, x_m$ increases the running-time required to compute a reduced Gröbner basis of the public key $\widetilde{\boldsymbol{E}} \in \boldsymbol{F}_2[x_1, \ldots, x_m]^l$

of the enhanced cryptosystem $\widetilde{\mathcal{K}}$. Thus, it would seem that the linear PH matrix method with random variables provides substantial robustness against the Gröbner basis attack.

In the above examples on the enhanced HFE by the PH method, due to the constraint of computing ability, only the cases of $p = 10$ and $l = 25, 28$ are computed where the ratios p/l of plain text to cipher text are $10/25 = 40\%$ and $10/28 \approx 36\%$, which seem to be inefficient. In realistic situations, however, p will usually be selected to be more than 100 and $l - p$ be $10 \sim 20$. Thus the ratio is not so inefficient in practice. We will continue to make examples for more large parameters.

4.3 Strength against Other Possible Attacks

As a countermeasure against the Gröbner basis attack, we appended additional variables to the public key of the enhanced cryptosystem $\widetilde{\mathcal{K}}$. However, a naive introduction of additional variables might allow another type of attack described below. In order to fend off such an attack against the enhanced cryptosystem $\widetilde{\mathcal{K}}$, we substituted the polynomial vector $A\boldsymbol{z}$ for partial variables in the original public key $\boldsymbol{E}$ of $\mathcal{K}$ in the right-hand side of (5).

It is not desirable that the eavesdropper, Catherine, can find the PH matrix M itself, or any matrix which works just like M, from a cipher text $\widetilde{\boldsymbol{c}}$ and the public key $(q, m, \widetilde{\boldsymbol{E}}, p)$ of $\widetilde{\mathcal{K}}$. This is because, if so, then she can easily obtain the value

$$\boldsymbol{E}\begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{d} \end{pmatrix}$$

due to the equation (7). However, in this PH matrix method with random variables, it would seem difficult to do so because of the existence of the vector $A\boldsymbol{z} \in \mathbf{F}_q[x_1, \ldots, x_m]^{k-p}$ substituted for the variables $x_{p+1}, \ldots, x_k$ in the original public key $\boldsymbol{E}$ of $\mathcal{K}$ in the right-hand side of (5).

Assume, contrarily to the fact, that $p = k$ and $A\boldsymbol{z}$ disappears from $\boldsymbol{E}$ in the right-hand side of (5). Then we have

$$\widetilde{\boldsymbol{E}} \equiv S\boldsymbol{E} + R\boldsymbol{Z}, \tag{8}$$

where $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ remains unchanged from the original public key of $\mathcal{K}$. In such a case, by trying to eliminate the variables $x_{k+1}, \ldots, x_m$ in $M'\widetilde{\boldsymbol{E}}$, Catherine may construct a matrix $M' \in {\mathbf{F}_q}^{n \times l}$ which satisfies that $M'\widetilde{\boldsymbol{E}} = \boldsymbol{E}$. This M' works in the same manner as the original PH matrix M, and therefore she may be able to calculate the value $\boldsymbol{E}(\boldsymbol{p}) \in {\mathbf{F}_q}^n$ from the cipher text vector $\widetilde{\boldsymbol{c}} \in {\mathbf{F}_q}^l$ defined by (6).

This possibility seems to be excluded by substituting $A\boldsymbol{z}$ for partial variables in $\boldsymbol{E}$ in the right-hand side of (5). This is because, since $A\boldsymbol{z}$ is a linear combination of all variables $x_1, \ldots, x_m$ in general, $\boldsymbol{E}$ in (5) also may contain all these variables and therefore Catherine loses the targets to eliminate. Thus, the attack by constructing a matrix M' which behaves just like the original PH matrix M may not be successful in this PH matrix method with random variables.

5 Nonlinearization of the PH Matrix

Another countermeasure against the Gröbner basis attack is to nonlinearize the PH matrix, i.e., to employ, as a PH matrix, a polynomial matrix N whose entries are in $\mathbf{F}_q[x_1, \ldots, x_k]$. Since an

ideal I generated by polynomials may change under the replacement of the generators of I by the product of N and them, the nonlinear PH matrix N may provide substantial robustness against the Gröbner basis attack, unlike in the case of the primitive implementation of a linear PH matrix method described in Subsection 3.1. In the previous work [22], we already proposed a nonlinear PH matrix method. However, the degree of polynomials in the public key of the enhanced cryptosystem $\widetilde{\mathcal{K}}$ is more than two in the previous method due to the use of Fermat's little theorem. This may cause an undesirable increase in the size of the public key of the enhanced cryptosystem. In this section, we propose a new nonlinear PH matrix method without using Fermat's little theorem, where the degree of polynomials in the public key of $\widetilde{\mathcal{K}}$ can be always the same as the degree of polynomials in the public key of the original cryptosystem $\mathcal{K}$. Thus, the new nonlinear PH matrix method is more practical than the previous proposal, from the point of view of the size of the public key. The new nonlinear method is described in what follows.

Let $\mathcal{K}$ be an arbitrary quadratic multivariate public key cryptosystem whose public key is given by $\boldsymbol{E} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$, as described in Subsection 2.2. We construct a new quadratic multivariate public key cryptosystem $\widetilde{\mathcal{K}}$ based on $\mathcal{K}$ as follows. Let l and h be any positive integers with $l \geq h$.

Key-Generation. In the key-generation stage, the public key and secret key of $\mathcal{K}$ are chosen first. Then, a public key $\widetilde{\boldsymbol{E}} \in \mathbf{F}_q[x_1, \ldots, x_k]^{l+h}$ of $\widetilde{\mathcal{K}}$ is constructed from the original public key $\boldsymbol{E}$ of $\mathcal{K}$ as follows. In the construction, a quadratic polynomial vector $\boldsymbol{C} \in \mathbf{F}_q[x_1, \ldots, x_k]^h$, a matrix $T \in {\mathbf{F}_q}^{n \times h}$, and a vector $\boldsymbol{u} \in {\mathbf{F}_q}^n$ are randomly chosen first. Then polynomial vectors $\boldsymbol{F} \in \mathbf{F}_q[x_1, \ldots, x_k]^n$ and $\overline{\boldsymbol{E}} \in \mathbf{F}_q[x_1, \ldots, x_k]^l$ are calculated in sequence by the following equations:

$$\begin{aligned} \boldsymbol{F} &\equiv \boldsymbol{E} - T\boldsymbol{C} - \boldsymbol{u}, \\ \overline{\boldsymbol{E}} &\equiv S \begin{pmatrix} 1 \\ \boldsymbol{F} \end{pmatrix} + R\boldsymbol{X}. \end{aligned}$$

Here $\boldsymbol{X}$ is defined in the same manner as in Subsection 3.1. S is a matrix in ${\mathbf{F}_q}^{l \times (n+1)}$. In order to make this nonlinear PH matrix method work properly, we assume that $l > n + 1$. On the other hand, R is a matrix in ${\mathbf{F}_q}^{l \times t}$. We also introduce the linear PH matrix $M \in {\mathbf{F}_q}^{(n+1) \times l}$ together with S and R. The matrices R, M, and S are randomly chosen in sequence so as to satisfy the following three conditions. This choice is efficiently possible, as in the case of the linear PH matrix methods in the previous sections.

Condition 4. $l \geq (n+1) + \operatorname{rank} R$. □

Condition 5. $MR = 0$ *and* $\operatorname{rank} M = n + 1$. □

Condition 6. $MS = I_{n+1}$, *where* I_{n+1} *is the identity matrix in* ${\mathbf{F}_q}^{(n+1) \times (n+1)}$. □

The nonlinear PH matrix $N \in \mathbf{F}_q[x_1, \ldots, x_k]^{(n+1) \times l}$ is then defined by

$$N \equiv (T\boldsymbol{C} + \boldsymbol{u} \quad I_n)M. \tag{9}$$

Finally, a public key $\widetilde{\boldsymbol{E}}$ of $\widetilde{\mathcal{K}}$ is calculated by the following equation:

$$\widetilde{\boldsymbol{E}} \equiv B \begin{pmatrix} \overline{\boldsymbol{E}} \\ \boldsymbol{C} \end{pmatrix}, \tag{10}$$

where B is a randomly chosen invertible matrix in $\mathbf{F}_q^{(l+h)\times(l+h)}$.

Then, the triple $(q, k, \widetilde{\boldsymbol{E}})$ forms the public key of $\widetilde{\mathcal{K}}$. Bob publishes it after the key-generation. On the other hand, B^{-1}, T, $\boldsymbol{u}$, and M, together with the secret key of $\mathcal{K}$ corresponding to the public key $(q, k, \boldsymbol{E})$ of $\mathcal{K}$, forms the secret key of $\widetilde{\mathcal{K}}$.

We here check that, by Conditions 5 and 6,

$$\begin{aligned} N\overline{\boldsymbol{E}} =& (T\boldsymbol{C}+\boldsymbol{u} \quad I_n)MS \begin{pmatrix} 1 \\ \boldsymbol{F} \end{pmatrix} + (T\boldsymbol{C}+\boldsymbol{u} \quad I_n)MR\boldsymbol{X} \\ =& (T\boldsymbol{C}+\boldsymbol{u} \quad I_n) \begin{pmatrix} 1 \\ \boldsymbol{F} \end{pmatrix} \\ =& T\boldsymbol{C}+\boldsymbol{u}+\boldsymbol{F} \\ =& \boldsymbol{E}. \end{aligned}$$

Hence we have

$$N\overline{\boldsymbol{E}} = \boldsymbol{E}, \tag{11}$$

and therefore N properly works as a PH matrix although it is a polynomial matrix. We can also check that the public key $\widetilde{\boldsymbol{E}}$ is certainly a quadratic polynomial vector as desired.

Encryption. A plain text of $\widetilde{\mathcal{K}}$ is represented by a vector in $\mathbf{F}_q^{\,k}$ in the same way as in $\mathcal{K}$. Now, assume that Alice wants to send Bob a plain text vector $\boldsymbol{p} \in \mathbf{F}_q^{\,k}$. The corresponding cipher text of $\widetilde{\mathcal{K}}$ is represented by a vector $\widetilde{\boldsymbol{c}} \in \mathbf{F}_q^{\,l+h}$, and is calculated through $\widetilde{\boldsymbol{c}} \equiv \widetilde{\boldsymbol{E}}(\boldsymbol{p})$ by Alice.

Decryption. The decryption of $\widetilde{\mathcal{K}}$ proceeds as follows. We first note, by (10), that

$$\begin{pmatrix} \overline{\boldsymbol{E}}(\boldsymbol{p}) \\ \boldsymbol{C}(\boldsymbol{p}) \end{pmatrix} = B^{-1}\widetilde{\boldsymbol{E}}(\boldsymbol{p}).$$

Thus, on receiving the cipher text $\widetilde{\boldsymbol{c}}$ corresponding to a plain text $\boldsymbol{p}$, Bob can efficiently obtain the values $\overline{\boldsymbol{E}}(\boldsymbol{p})$ and $\boldsymbol{C}(\boldsymbol{p})$ from the multiplication of $\widetilde{\boldsymbol{c}}$ by B^{-1} from the left. Since

$$N(\boldsymbol{p}) = (T\boldsymbol{C}(\boldsymbol{p})+\boldsymbol{u} \quad I_n)M$$

by (9), Bob can then calculate $N(\boldsymbol{p})$ using the value $\boldsymbol{C}(\boldsymbol{p})$ and the secret key of $\widetilde{\mathcal{K}}$. It follows from (11) that

$$N(\boldsymbol{p})\overline{\boldsymbol{E}}(\boldsymbol{p}) = \boldsymbol{E}(\boldsymbol{p}).$$

Thus, using the values $N(\boldsymbol{p})$ and $\overline{\boldsymbol{E}}(\boldsymbol{p})$, Bob can efficiently calculate the value $\boldsymbol{E}(\boldsymbol{p})$. Then, according to the decryption procedure of $\mathcal{K}$, Bob can finally recover the plain text $\boldsymbol{p}$ using the secret key of $\mathcal{K}$.

Remark 5.1. In the above, for simplicity, we only describe the nonlinear PH matrix method in a primitive form. For the purpose of enhancing the security further, it is possible to construct a nonlinear PH matrix method with random variables in the same manner as we made changes and improvements in the primitive linear PH matrix method described in Subsection 3.1 to obtain the linear PH matrix method with random variables in Subsection 4.1. Thus, the resulting nonlinear PH matrix method with random variables might provide more security against the Gröbner basis attack than the linear PH matrix method with random variables.

6 Concluding Remarks

In this paper, we have elaborated the piece in hand (PH) matrix methods in order that the security of a wide class of multivariate public key cryptosystems is likely to be enhanced by them even against the Gröbner basis attack. In the future work, we will demonstrate the enhancement of security both by the linear PH matrix method with random variables (Section 4) and by the nonlinear PH matrix method (Section 5) against the Gröbner basis attack in an experimental manner for all proposed multivariate public key cryptosystems extensively.

Besides the Gröbner basis attack, several attacks are proposed so far against multivariate public key cryptosystems in general. For example, Fouque, Granboulan, and Stern recently adapted differential cryptanalysis to multivariate public key cryptosystems in [6]. In the linear PH matrix method with random variables described in Section 4, the public key of the enhanced cryptosystem $\widetilde{\mathcal{K}}$ has additional variables. Thus, it might be an effective attack against this PH matrix method to reduce the number of variables by fixing some of these additional variables with appropriate values prior to computing a Gröbner basis of the public key (see [1] for this attack). The effectiveness of these attacks against our PH matrix methods proposed in this paper will be considered in the future work.

From the practical point of view, it is also important to evaluate the key length and the efficiency of encryption and decryption in the enhanced cryptosystem. However, since the aim of the present paper is mainly to improve the framework of PH matrix method, this issue is discussed in another paper. Because of the same reason, we have not considered the stronger security such as IND-CCA type security but considered just the encryption primitive $\widetilde{\boldsymbol{E}}$ for a multivariate public key cryptosystem whose security is enhanced by the PH matrix method. We leave the consideration of the stronger security to a future study.

Acknowledgments

The authors are grateful to Mr. Gwénolé Ars for discussions and helpful comments. Especially, he pointed out that the relation (4) holds for the original linear PH matrix method proposed previously.

References

[1] N. Courtois, M. Daum, and P. Felke. On the security of HFE, HFEv- and Quartz. *Proc. PKC 2003*, Lecture Notes in Computer Science, Vol.2567, pp.337–350, Springer, 2003.

[2] N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. *Proc. EUROCRYPT 2000*, Lecture Notes in Computer Science, Vol.1807, pp.392–407, Springer, 2000.

[3] A. Kipnis, J. Patarin, and L. Goubin. Unbalanced Oil and Vinegar signature schemes. *Proc. EUROCRYPT 1999*, Lecture Notes in Computer Science, Vol.1592, pp.206–222, Springer, 1999.

[4] J. Ding. A new variant of the Matsumoto-Imai cryptosystem through perturbation. *Proc. PKC 2004*, Lecture Notes in Computer Science, Vol.2947, pp.305–318, Springer, 2004.

[5] J. C. Faugère and A. Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases. *Proc. CRYPTO 2003*, Lecture Notes in Computer Science, Vol.2729, pp.44–60, Springer, 2003.

[6] P. A. Fouque, L. Granboulan, and J. Stern. Differential cryptanalysis for multivariate schemes. *Proc. EUROCRYPT 2005*, Lecture Notes in Computer Science, Vol.3494, pp.341–353, Springer, 2005.

[7] S. Hasegawa and T. Kaneko. An attacking method for a public-key cryptosystem based on the difficulty of solving a system of non-linear equations. *Proc. 10th SITA*, JA5-3, November 1987. In Japanese.

[8] M. Kasahara and R. Sakai. A new principle of public key cryptosystem and its realization. Technical Report of IEICE, ISEC2000-92 (2000-11), November 2000. In Japanese.

[9] M. Kasahara and R. Sakai. A construction of public key cryptosystem for realizing ciphertext of size 100 bit and digital signature scheme. *IEICE Transactions*, E87-A, No.1 (2004), 102–109.

[10] M. Kasahara and R. Sakai. A construction of publickey cryptosystem based on singular simultaneous equations. *Proc. SCIS2004*, 1B5-1, 2004.

[11] M. Kasahara and R. Sakai. A construction of public-key cryptosystem based on singular simultaneous equations and its variants. Technical Report of IEICE, ISEC2005-7 (2005-05), May 2005.

[12] A. Kipnis and A. Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. *Proc. CRYPTO '99*, Lecture Notes in Computer Science, Vol.1666, pp.19–30, Springer, 1999.

[13] T. Matsumoto, H. Imai, H. Harashima, and H. Miyakawa. A class of asymmetric cryptosystems using obscure representations of enciphering functions. *1983 National Convention Record on Information Systems, IECE Japan*, S8-5, 1983. In Japanese.

[14] T. Matsumoto and H. Imai. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. *Proc. EUROCRYPT '88*, Lecture Notes in Computer Science, Vol.330, pp.419–453, Springer, 1988.

[15] T. T. Moh. A public key system with signature and master key functions. *Communications in Algebra*, 27, 2207–2222, 1999.

[16] J. Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt '88. *Proc. CRYPTO '95*, Lecture Notes in Computer Science, Vol.963, pp.248-261, Springer, 1995.

[17] J. Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. *Proc. EUROCRYPT '96*, Lecture Notes in Computer Science, Vol.1070, pp.33–48, Springer, 1996.

[18] S. Tsujii, K. Kurosawa, T. Itoh, A. Fujioka, and T. Matsumoto. A public-key cryptosystem based on the difficulty of solving a system of non-linear equations. *IEICE Transactions* (D), J69-D, No.12 (1986), 1963–1970. In Japanese.

[19] S. Tsujii, A. Fujioka, and Y. Hirayama. Generalization of the public-key cryptosystem based on the difficulty of solving a system of non-linear equations. *IEICE Transactions* (A), J72-A, No.2 (1989), 390–397. In Japanese.

[20] S. Tsujii. A new structure of primitive public key cryptosystem based on soldiers in hand matrix. Technical Report TRISE 02-03, Chuo University, July 2003.

[21] S. Tsujii, R. Fujita, and K. Tadaki. Proposal of MOCHIGOMA (piece in hand) concept for multivariate type public key cryptosystem. Technical Report of IEICE, ISEC2004-74 (2004-09), September 2004.

[22] S. Tsujii, K. Tadaki, and R. Fujita. Piece in hand concept for enhancing the security of multivariate type public key cryptosystems: public key without containing all the information of secret key. Cryptology ePrint Archive, Report 2004/366, December 2004. Available at URL: `http://eprint.iacr.org/2004/366` .

[23] S. Tsujii, K. Tadaki, and R. Fujita. Piece in hand concept for enhancing the security of multivariate type public key cryptosystems: public key without containing all the information of secret key. *Proc. SCIS2005*, 2E1-3, pp.487–492, 2005.

[24] S. Tsujii, K. Tadaki, and R. Fujita. Proposal for piece in hand (soldiers in hand) matrix — general concept for enhancing security of multivariate public key cryptosystems — Ver.2. *Proc. SCIS2006*, 2A4-1, 2006. In Japanese.

[25] C. Wolf, A. Braeken, and B. Preneel. Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC. *Proc. SCN 2004*, Lecture Notes in Computer Science, Vol.3352, pp.294–309, Springer, 2004.

[26] C. Wolf and B. Preneel. Taxonomy of Public Key Schemes based on the problem of Multivariate Quadratic equations. Cryptology ePrint Archive, Report 2005/077, December 2005. Available at URL: `http://eprint.iacr.org/2005/077` .

Post-quantum lattice-based cryptography

Phong Nguyen

École Nationale Supérieure

NTRUEncrypt and NTRUSign: efficient public key algorithms for a post-quantum world

Jeff Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, William Whyte

NTRU Cryptosystems,
35 Nagog Park, Acton, MA 01720, USA
{jhoffstein,nhowgravegraham,jpipher,jsilverman,wwhyte}@ntru.com

Abstract. We provide an overview of the algorithms NTRUEncrypt and NTRUSign. These algorithms have attractive operating speed and keysize and are based on hard problems that are not known to have polynomial-time quantum algorithms. We discuss the state of current knowledge about the security of both algorithms and identify areas of research where progress could help the algorithms to gain acceptance.

1 Introduction

In considering what cryptographic techniques may be of use if practical quantum computers are ever developed, it is natural to look at algorithms based on lattice reduction. Lattice reduction is a reasonably well-studied hard problem that is currently not known to be solved by any polynomial time, or even subexponential time, quantum algorithms [37, 27]. The NTRUEncrypt and NTRUSign algorithms are currently the most attractive public key algorithms whose security rests on lattice reduction: in common with other lattice-based algorithms, they have fast running times (typically $O(k^2)$ for security levels of k bits), but unlike other lattice-based algorithms the public keys, signatures, and ciphertexts are of size $O(k)$ rather than $O(k^2)$. In fact, at the 112-bit security level and above, NTRUSign public keys are actually smaller than equivalent-strength RSA keys. Published performance figures for NTRUEncrypt and NTRUSign show them to be among the fastest public-key algorithms currently known. The algorithms are therefore of interest even in the classical computing world, and are clearly prime candidates for widespread adoption should quantum computers ever be invented.

This paper presents an overview of operations, performance, and security considerations for both algorithms. The most up-to-date descriptions of NTRUEncrypt and NTRUSign are included in [21] and [20], respectively. This paper summarizes, and draws heavily on, the material presented in those papers.

This paper is structured as follows. First, we introduce and describe the algorithms NTRUEncrypt and NTRUSign. We then survey known results about the security of these algorithms, including their security against known quantum algorithms. We then present performance characteristics of the algorithms.

2 NTRUEncrypt: Overview

2.1 Parameters and Definitions

An implementation of the NTRUEncrypt encryption primitive is specified by the following parameters:

N	*Degree Parameter.* A positive integer. The associated NTRU lattice has dimension $2N$.
q	*Large Modulus.* A positive integer. The associated NTRU lattice is a convolution modular lattice of modulus q.
p	*Small Modulus.* An integer or a polynomial.
$\mathcal{D}_f, \mathcal{D}_g$	*Private Key Spaces.* Sets of small polynomials from which the private keys are selected.
$\mathcal{D}_m$	*Plaintext Space.* Set of polynomials that represent encryptable messages. It is the responsibility of the encryption scheme to provide a method for encoding the message that one wishes to encrypt into a polynomial in this space.
$\mathcal{D}_r$	*Blinding Value Space.* Set of polynomials from which the temporary blinding value used during encryption is selected.

`center` *Centering Method.* A means of performing mod q reduction on decryption.

Definition 1. *The* Ring of Convolution Polynomials *is*

$$\mathcal{R} = \frac{\mathbb{Z}[X]}{(X^N - 1)}.$$

Multiplication of polynomials in this ring corresponds to the convolution product of their associated vectors, defined by

$$(\mathsf{f} * \mathsf{g})(X) = \sum_{k=0}^{N-1} \left(\sum_{i+j \equiv k \pmod N} \mathsf{f}_i \cdot \mathsf{g}_j \right) X^k \ .$$

We also use the notation $\mathcal{R}_q = \frac{(\mathbb{Z}/q\mathbb{Z})[X]}{(X^N-1)}$. *Convolution operations in the ring* $\mathcal{R}_q$ *are referred to as* modular convolutions.

Definition 2. *A polynomial* $a(X) = a_0 + a_1 X + \cdots + a_{N-1}X^{N-1}$ *is identified with its vector of coefficients* $\mathsf{a} = [a_0, a_1, \ldots, a_{N-1}]$. *The mean* $\bar{\mathsf{a}}$ *of a polynomial* a *is defined by* $\bar{a} = \frac{1}{N}\sum_{i=0}^{N-1} \mathsf{a}_i$. *The* centered norm $\|\mathsf{a}\|$ *of* a *is defined by*

$$\|\mathsf{a}\|^2 = \sum_{i=0}^{N-1} a_i^2 - \frac{1}{N}\left(\sum_{i=0}^{N-1} a_i\right)^2 . \tag{1}$$

Definition 3. *The* width $\mathsf{Width}(\mathsf{a})$ *of a polynomial or vector is defined by*

$$\mathsf{Width}(\mathsf{a}) = \mathsf{Max}(a_0, \ldots, a_{N-1}) - \mathsf{Min}(a_0, \ldots, a_{N-1}) \ .$$

Definition 4. *A* binary polynomial *is one whose coefficients are all in the set* $\{0, 1\}$. *A* trinary polynomial *is one whose coefficients are all in the set* $\{0, \pm 1\}$. *If one of the inputs to a convolution is a binary polynomial, the operation is referred to as a* binary convolution. *If one of the inputs to a convolution is a trinary polynomial, the operation is referred to as a* trinary convolution.

Definition 5. *Define the polynomial spaces* $\mathcal{B}_N(d), \mathcal{T}_N(d), \mathcal{T}_N(d_1, d_2)$ *as follows. Polynomials in* $\mathcal{B}_N(d)$ *have* d *coefficients equal to* 1 *and the other coefficients are* 0. *Polynomials in* $\mathcal{T}_N(d)$ *have* $d+1$ *coefficients equal to* 1, *have* d *coefficients equal to* -1, *and the other coefficients are* 0. *Polynomials in* $\mathcal{T}_N(d_1, d_2)$ *have* d_1 *coefficients equal to* 1, *have* d_2 *coefficients equal to* -1, *and the other coefficients are* 0.

2.2 "Raw" NTRUEncrypt

Key Generation NTRUEncrypt **key generation** consists of the following operations:

1. Randomly generate polynomials f and g in $\mathcal{D}_f$, $\mathcal{D}_g$ respectively.
2. Invert f in $\mathcal{R}_q$ to obtain f_q, invert f in $\mathcal{R}_p$ to obtain f_p, and check that g is invertible in $\mathcal{R}_q$ [14].
3. The public key $\mathsf{h} = p * \mathsf{g} * \mathsf{f}_q \pmod q$. The private key is the pair $(\mathsf{f}, \mathsf{f}_p)$.

Encryption NTRUEncrypt **encryption** consists of the following operations:

1. Randomly select a "small"polynomial $\mathsf{r} \in \mathcal{D}_r$.
2. Calculate the ciphertext e as $\mathsf{e} \equiv \mathsf{r} * \mathsf{h} + \mathsf{m} \pmod q$.

Decryption NTRUEncrypt **decryption** consists of the following operations:

1. Calculate $\mathsf{a} \equiv \mathtt{center}(\mathsf{f} * e)$, where the centering operation `center` reduces its input into the interval $[A, A + q - 1]$.
2. Recover m by calculating $\mathsf{m} \equiv \mathsf{f}_p * \mathsf{a} \pmod p$.

To see why decryption works, use $\mathsf{h} \equiv p * \mathsf{g} * \mathsf{f}_q$ and $\mathsf{e} \equiv \mathsf{r} * \mathsf{h} + \mathsf{m}$ to obtain

$$\mathsf{a} \equiv p * \mathsf{r} * \mathsf{g} + \mathsf{f} * \mathsf{m} \pmod q \ . \tag{2}$$

For appropriate choices of parameters and `center`, this is an equality over $\mathbb{Z}$, rather than just over $\mathbb{Z}_q$. Therefore step 2 recovers m: the $p * \mathsf{r} * \mathsf{g}$ term vanishes, and $\mathsf{f}_p * \mathsf{f} * \mathsf{m} = \mathsf{m} \pmod p$.

2.3 Encryption schemes: NAEP

In order to protect against adaptive chosen ciphertext attacks, we must use an appropriately defined *encryption scheme*. The scheme described in [18] gives provable security in the random oracle model [2, 3] with a tight (ie linear) reduction. We briefly outline it here.

NAEP uses two hash functions:

$$G : \{0,1\}^{N-l} \times \{0,1\}^l \to \mathcal{D}_r \quad H : \{0,1\}^N \to \{0,1\}^N$$

To encrypt a message $M \in \{0,1\}^{N-l}$ using NAEP one uses the functions

$$\texttt{compress}(x) = (x \pmod q) \pmod 2,$$
$$\texttt{B2P} : \{0,1\}^N \to \mathcal{D}_m \cup \text{“error”}, \quad \texttt{P2B} : \mathcal{D}_m \to \{0,1\}^N$$

The function `compress` puts the coefficients of the modular quantity $x \pmod q$ in to the interval $[0, q)$, and then this quantity is reduced modulo 2. The role of `compress` is simply to reduce the size of the input to the hash function H for gains in practical efficiency.The function `B2P` converts a bit string into a binary polynomial, or returns “error” if the bit string does not fulfil the appropriate criteria – for example, if it does not have the appropriate level of combinatorial security. The function `P2B` converts a binary polynomial to a bit string.

The encryption algorithm is then specified by:

1. Pick $b \xleftarrow{R} \{0,1\}^l$.
2. Let $\mathsf{r} = G(M, b)$, $\mathsf{m} = \texttt{B2P}(\ (M||b) \oplus H(\texttt{compress}(\mathsf{r} * \mathsf{h}))\ )$.
3. If `B2P` returns “error”, go to step 1.
4. Let $\mathsf{e} = \mathsf{r} * \mathsf{h} + \mathsf{m} \in \mathcal{R}_q$.

Step 3 ensures that only messages of the appropriate form will be encrypted.

To decrypt a message $\mathsf{e} \in \mathcal{R}_q$ one does the following:

1. Let $\mathsf{a} = \texttt{center}(\mathsf{f} * \mathsf{e} \pmod q)$.
2. Let $\mathsf{m} = \mathsf{f}_p^{-1} * \mathsf{a} \pmod p$.
3. Let $\mathsf{s} = \mathsf{e} - \mathsf{m}$.
4. Let $M||b = \texttt{P2B}(\mathsf{m}) \oplus H(\texttt{compress}(\texttt{P2B}(\mathsf{s})))$.
5. Let $\mathsf{r} = G(M, b)$.
6. If $\mathsf{r} * \mathsf{h} = \mathsf{s} \pmod q$, and $\mathsf{m} \in \mathcal{D}_m$, then return the message M, else return the string “invalid ciphertext”.

The use of the scheme NAEP introduces a single additional parameter:

- l — *Random Padding Length.* The length of the random padding b concatenated with M in step 1.

2.4 Instantiating NAEP: SVES-3

The EESS#1 v2 standard [5] specifies an instantiation of NAEP known as SVES-3. In SVES-3, the following specific design choices are made:

- To allow variable-length messages, a one-byte encoding of the message length in bytes is prepended to the message. The message is padded with zeroes to fill out the message block.
- The hash function G which is used to produce r takes as input M; b; an OID identifying the encryption scheme and parameter set; and a string h_{trunc} derived by truncating the public key to length l_h bits.

SVES-3 includes h_{trunc} in G so that r depends on the specific public key. Even if an attacker were to find an (M, b) that gave an r with an increased chance of a decryption failure, that (M, b) would apply only to a single public key and could not be used to attack other public keys. However, the current recommended parameter sets do not have decryption failures and so there is no need to input h_{trunc} to G. We will therefore use SVES-3but set $l_h = 0$.

3 NTRUSign: Overview

3.1 Parameters

An implementation of the NTRUSign primitive uses the following parameters:

N	polynomials have degree $< N$
q	coefficients of polynomials are reduced modulo q
$\mathcal{D}_f, \mathcal{D}_g$	polynomials in $\mathcal{T}(d)$ have $d+1$ coefficients equal to 1, have d coefficients equal to -1, and the other coefficients are 0.
$\mathcal{N}$	the norm bound used to verify a signature.
β	the balancing factor for the norm $\|\cdot\|_\beta$. Has the property $0 < \beta \leq 1$.

3.2 "Raw" NTRUSign

Key Generation NTRUSign **key generation** consists of the following operations:

1. Randomly generate "small" polynomials f and g in $\mathcal{D}_f$, $\mathcal{D}_g$ respectively such that f and g are invertible modulo q.
2. Find polynomials F and G such that
$$\mathsf{f} * \mathsf{G} - \mathsf{g} * \mathsf{F} = q \ , \tag{3}$$
and F and G have size
$$\|\mathsf{F}\| \approx \|\mathsf{G}\| \approx \|\mathsf{f}\|\sqrt{N/12} \ . \tag{4}$$
This can be done using the methods of [19]
3. Denote the inverse of f in $\mathcal{R}_q$ by f_q, and the inverse of g in $\mathcal{R}_q$ by g_q The public key $\mathsf{h} = \mathsf{F} * \mathsf{f}_q \pmod q = \mathsf{G} * \mathsf{g}_q \pmod q$. The private key is the pair (f, g).

Signing The signing operation involves *rounding* polynomials. For any $a \in \mathbb{Q}$, let $\lfloor a \rceil$ denote the integer closest to a, and define $\{a\} = a - \lfloor a \rceil$. (For numbers a that are midway between two integers, we specify that $\{a\} = +\frac{1}{2}$, rather than $-\frac{1}{2}$.) If A is a polynomial with rational (or real) coefficients, let $\lfloor A \rceil$ and $\{A\}$ be A with the indicated operation applied to each coefficient.

"Raw" NTRUSign **signing** consists of the following operations:

1. Map the digital document D to be signed to a vector $\mathsf{m} \in [0, q)^N$ using an agreed hash function.
2. Set
$$(\mathsf{x}, \mathsf{y}) = (0, \mathsf{m}) \begin{pmatrix} \mathsf{G} & -\mathsf{F} \\ -\mathsf{g} & \mathsf{f} \end{pmatrix} / q = \left(\frac{-\mathsf{m} * \mathsf{g}}{q}, \frac{\mathsf{m} * \mathsf{f}}{q} \right) .$$
3. Set
$$\epsilon = -\{x\} \qquad \text{and} \qquad \epsilon' = -\{y\} \ . \tag{5}$$
4. Calculate s, the signature, as
$$\mathsf{s} = \epsilon \mathsf{f} + \epsilon' \mathsf{g} \ . \tag{6}$$

Verification Verification involves the use of a *balancing factor* β and a *norm bound* $\mathcal{N}$. To verify, the recipient does the following:

1. Map the digital document D to be verified to a vector $\mathsf{m} \in [0, q)^N$ using the agreed hash function.
2. Calculate $\mathsf{t} = \mathsf{s} * \mathsf{h} \bmod q$, where s is the signature and h is the signer's public key.
3. Calculate the norm
$$\nu = \min_{k_1, k_2 \in R} \left(\|\mathsf{s} + k_1 q\|^2 + \beta^2 \|(\mathsf{t} - \mathsf{m}) + k_2 q\|^2 \right)^{1/2} \ . \tag{7}$$
4. If $\nu \leq \mathcal{N}$, the verification succeeds. Otherwise, it fails.

3.3 Why NTRUSign works

Given any positive integers N and q and any polynomial $h \in R$, we can construct a lattice L_h contained in $R^2 \cong \mathbb{Z}^{2N}$ as follows:

$$L_h = L_h(N, q) = \big\{(r, r') \in R \times R \;\big|\; r' \equiv r * h \pmod q\big\}.$$

This sublattice of $\mathbb{Z}^{2N}$ is called a *convolution modular lattice*. It has dimension equal to $2N$ and determinant equal to q^N.

Since

$$\det \begin{pmatrix} \mathsf{f} & \mathsf{F} \\ \mathsf{g} & \mathsf{G} \end{pmatrix} = q$$

and we have defined $\mathsf{h} = \mathsf{F}/\mathsf{f} = \mathsf{G}/\mathsf{g} \bmod q$, we know that

$$\begin{pmatrix} \mathsf{f} & \mathsf{F} \\ \mathsf{g} & \mathsf{G} \end{pmatrix} \text{ and } \begin{pmatrix} 1 & \mathsf{h} \\ 0 & q \end{pmatrix}$$

are bases for the same lattice. Here, as in [19], a 2-by-2 matrix of polynomials is converted to a $2N$-by-$2N$ integer matrix matrix by converting each polynomial in the polynomial matrix to its representation as an N-by-N circulant matrix, and the two representations are regarded as equivalent.

Signing consists of finding a close lattice point to the message point $(0, \mathsf{m})$ using Babai's method: express the target point as a real-valued combination of the basis vectors, and find a close lattice point by rounding off the fractional parts of the real coefficients to obtain integer combinations of the basis vectors. The error introduced by this process will be the sum of the rounding errors on each of the basis vectors, and the rounding error will by definition be between $-\frac{1}{2}$ and $\frac{1}{2}$. In NTRUSign, the basis vectors are all of the same length, so the expected error introduced by $2N$ roundings of this type will be $\sqrt{N/6}$ times this length.

In NTRUSign, the private basis is chosen such that $\|\mathsf{f}\| = \|\mathsf{g}\|$ and $\|\mathsf{F}\| \sim \|\mathsf{G}\| \sim \sqrt{N/12}\|\mathsf{f}\|$. The expected error in signing will therefore be

$$\sqrt{N/6}\|\mathsf{f}\| + \beta(N/6\sqrt{2})\|\mathsf{f}\|. \tag{8}$$

In contrast, an attacker who uses only the public key will likely produce a signature with N incorrect coefficients, and those coefficients will be distributed randomly mod q. The expected error in generating a signature with a public key is therefore

$$\beta\sqrt{N/12}q\ . \tag{9}$$

(We discuss security considerations in more detail in Section 9 and onwards; the purpose of this section is to argue that it is plausible that the private key allows the production of smaller signatures than the public key).

It is therefore clear that it is possible to choose $\|\mathsf{f}\|$ and q such that knowledge of the private basis allows the creation of smaller signing errors than knowledge of the public basis alone. Therefore, by ensuring that the signing error is less than could be expected to be produced by the public basis, a recipient can verify that the signature was produced by the owner of the private basis and is therefore valid.

3.4 NTRUSign signature schemes: chosen message attacks, hashing and message preprocessing

To prevent chosen message attacks the message representative m must be generated in some pseudo-random fashion from the input document D. The currently recommended hash function for NTRUSign is a simple Full Domain Hash. First the message is hashed to a "seed" hash value H_m. H_m is then hashed in counter mode to produce the appropriate number of bits of random output, which are treated as N numbers mod q. Since q is a power of 2, there are no concerns with bias.

The above mechanism is deterministic. If parameter sets were chosen that gave a significant chance of signature failure, the mechanism can be randomized as follows. The additional input to the process is r_{len}, the length of the randomizer in bits.

On signing:

1. Hash the message as before to generate H_m.

2. Select a randomizer r consisting of r_{len} random bits.
3. Hash $H_m \| r$ in counter mode to obtain enough output for the message representative m.
4. On signing, check that the signature will verify correctly.
   (a) If the signature does not verify, repeat the process with a different r.
   (b) If the signature verifies, send the tuple (r, s) as the signature

On verification, the verifier uses the received r and the calculated H_m as input to the hash in counter mode to generate the same message representative as the signer used.

The size of r should be related to the probability of signature failure. An attacker who is able to determine through timing information that a given H_m required multiple rs knows that at least one of those rs resulted in a signature that was too big, but does not know which message it was or what the resulting signature was. It is an open research question to quantify the appropriate size of r for a given signature failure probability, but in most cases $r_{\text{len}} = 8$ or 32 should be sufficient.

3.5 NTRUSign signature schemes: perturbations

To protect against transcript attacks, the raw NTRUSign signing algorithm defined above is modified as follows.

On key generation, the signer generates a secret *perturbation distribution function.*

On signing, the signer uses the agreed hash function to map the document D to the message representative m. However, before using her private key, she chooses an error vector e drawn from the perturbation distribution function that was defined as part of key generation. She then signs $\mathsf{m} + \mathsf{e}$, rather than m alone.

The verifier calculates m, t, and the norms of s and $\mathsf{t} - \mathsf{m}$ and compares the norms to a specified bound $\mathcal{N}$ as before. Since signatures with perturbations will be larger than unperturbed signatures, $\mathcal{N}$ and in fact all of the parameters will in general be different for the perturbed and unpertubed cases.

NTRU currently recommends the following mechanism for generating perturbations.

Key generation At key generation time, the signer generates B lattices $L_1 \dots L_B$. These lattices are generated with the same parameters as the private and public key lattice, L_0, but are otherwise independent of L_0 and of each other. For each L_i, the signer stores f_i, g_i, h_i.

Signing When signing m, for each L_i starting with L_B, the signer does the following:

1. Set $(\mathsf{x}, \mathsf{y}) == \left(\frac{-\mathsf{m} * \mathsf{g}_i}{q}, \frac{\mathsf{m} * \mathsf{f}_i}{q}\right)$.
2. Set $\epsilon = -\{x\}$ and $\epsilon' = -\{y\}$.
3. Set $\mathsf{s}_i = \epsilon \mathsf{f}_i + \epsilon' \mathsf{g}_i$.
4. Set $\mathsf{s} = \mathsf{s} + \mathsf{s}_i$.
5. If $i = 0$ stop and output s; otherwise, continue
6. Set $\mathsf{t}_i = \mathsf{s}_i * \mathsf{h}_i \bmod q$
7. Set $\mathsf{m} = \mathsf{t}_i - (\mathsf{s}_i * \mathsf{h}_{i-1}) \bmod q$.

The final step translates back to a point of the form $(0, \mathsf{m})$ so that all the signing operations can use only the f and g components, allowing for greater efficiency. Note that steps 6 and 7 can be combined into the single step of setting $\mathsf{m} = \mathsf{s}_i * (\mathsf{h}_i - \mathsf{h}_{i-1}$ to improve performance.

The parameter sets defined in [20] take $B = 1$.

4 NTRUEncrypt performance

4.1 NTRUEncrypt parameter sets

There are many different ways of choosing "small" polynomials. This section reviews NTRU's current recommendations for choosing the form of these polynomials for best efficiency. We focus here on choices that improve efficiency; security considerations are looked at in Section 8.

Form of f Published NTRUEncrypt parameter sets [21] take f to be of the form $\mathsf{f} = 1 + \mathsf{pF}$. This guarantees that $\mathsf{f}_p = 1$, eliminating one convolution on decryption.

Form of F, g, r NTRU currently recommends two different forms for F and R. If F and r take *binary* form, they are drawn from $\mathcal{B}_N(d)$, the set of binary polynomials with d 1s and $N - d$ 0s. If F and r take *product* form, then $\mathsf{F} = \mathsf{f}_1 * \mathsf{f}_2 + \mathsf{f}_3$, with $\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3 \stackrel{R}{\leftarrow} \mathcal{B}_N(d)$, and similarly for r. (The value d is considerably lower in the product-form case than in the binary case). In both cases, it turns out that the best security and performance is obtained by taking $\mathsf{g} \stackrel{R}{\leftarrow} \mathcal{B}_N(\lfloor N/2 \rfloor)$.

A binary convolution requires dN adds mod q. The best efficiency is therefore obtained when d is as low as possible consistent with the security requirements.

Plaintext size For k-bit security, we want to transport $2k$ bits of message and we we require $l \geq k$, l the random padding length. SVES-3 uses 8 bits to encode the length of the transported message. N must therefore be at least $3k + 8$. Smaller N will in general lead to lower bandwidth and faster operations.

Form of p, q The parameters p and q must be relatively prime. This admits of various combinations, such as $(p = 2, q = \text{prime})$, $(p = 3, q = 2^m)$, $(p = 2 + X, q = 2^m)$, The only combination that allows $q < 256$ for typical security levels without creating the possibility of decryption failures is $(p = 2, q = \text{prime})$. The current parameter sets therefore take p and q to be of this form.

The B2P function The polynomial m produced by the B2P function will be a random binary polynomial. As the number of 1s (or 0s) diverges from $N/2$, the strength of the ciphertext against both lattice and combinatorial attacks will decrease. The B2P function therefore contains a check that the number of 1s in m is no less than a value d_{m_0}. This value is chosen such that the chance that a randomly chosen polynomial lies outside this range is no more than 2^{-40}.

4.2 NTRUEncrypt performance

Table 1 and Table 2 give parameter sets and running times (in terms of operations per second) for binary and product-form cases respectively at different security levels corresponding to k bits of security. "Size" is the size of the public key in bits. In the case of NTRUEncrypt and RSA this is also the size of the ciphertext; in the case of some ECC encryption schemes, such as ECIES, the ciphertext may be a multiple of this size. Times given are for unoptimized C implementations on a 1.7 GHz Pentium and include time for all encryption scheme operations, including hashing, random number generation, as well as the primitive operation. d_{m_0} is the same in both the binary and product-form case and is omitted from the product-form table.

For comparison, we provide the times given in [4] for raw elliptic curve point multiplication (not including hashing or random number generation times) over the NIST prime curves. These times were obtained on a 400 MHz SPARC and have been converted to operations per second by simply scaling by 400/1700. Times given are for point multiplication without precomputation, as this corresponds to common usage in encryption and decryption. Precomputation improves the point multiplication times by a factor of 3.5-4. We also give the speedup for NTRUEncrypt decryption versus a single ECC point multiplication.

5 NTRUSign performance

5.1 NTRUSign parameter sets

Form of f, g The current recommended parameter sets take f and g to be trinary, i.e. drawn from $\mathcal{T}_N(d)$. Trinary polynomials allow for higher combinatorial security than binary polynomials at a given value of N and admit of efficient implementations. A trinary convolution requires $(2d + 1)N$ adds and one subtract mod q. The best efficiency is therefore obtained when d is as low as possible consistent with the security requirements.

k	N	d	d_{m0}	q	size	RSA size	ECC size	enc/s	dec/s	ECC mult/s	Speedup wrt ECC
80	251	48	70	197	2008	1024	163	21607	11558	1632	7.08
112	347	66	108	269	3033	~ 2048	224	10449	5805	1075	5.4
128	397	74	128	307	3501	3072	256	8198	4729	661	7.15
160	491	91	167	367	4383	4096	320	5427	3078	—	—
192	587	108	208	439	5193	7680	384	3985	2263	196	11.55
256	787	140	294	587	7690	15360	512	2574	1416	115	12.31

Table 1. Final Parameter Sets for different values of k using binary polynomials.

k	N	d	q	size	RSA size	ECC size	Speedup wrt binary	Speedup wrt ECC
80	251	8	293	2259	1024	163	2.00	18.63
112	347	11	541	3370	~ 2048	224	2.00	14.88
128	397	12	659	3890	3072	256	2.06	19.40
160	491	15	967	4870	4096	320	2.02	29.22
192	587	17	1229	6347	7680	384	2.12	31.29
256	787	22	2027	8459	15360	512	2.12	30.72

Table 2. Final Parameter Sets for different values of k using product form polynomials.

Form of p, q The parameters q and N must be relatively prime. For efficiency, we take q to be a power of 2.

Signing Failures A low value of $\mathcal{N}$, the norm bound, gives the possibility that a validly generated signature will fail. This affects efficiency, as if the chance of failure is non-negligible the signer must randomize the message before signing and check for failure on signature generation. For efficiency, we want to set $\mathcal{N}$ sufficiently high to make the chance of failure negligible. To do this, we denote the expected size of a signature by $\mathcal{E}$ and define the *signing tolerance* ρ by the formula

$$\mathcal{N} = \rho\mathcal{E} \ .$$

As ρ increases beyond 1, the chance of a signing failure appears to drop off exponentially. In particular, experimental evidence indicates that the probability that a validly generated signature will fail the normbound test with parameter ρ is smaller than $e^{-C(N)(\rho-1)}$, where $C(N) > 0$ increases with N. In fact, under the assumption that each coefficient of a signature can be treated as a sum of independent identically distributed random variables, a theoretical analysis indicates that $C(N)$ grows quadratically in N. The parameter sets below were generated with $\rho = 1.1$, which appears to give a vanishingly small probability of valid signature failure for N in the ranges that we consider. It is an open research question to determine precise signature failure probabilities for specific parameter sets, i.e. to determine the constants in $C(N)$.

5.2 NTRUSign performance

With one perturbation, signing takes time equivalent to two "raw" signing operations (as defined in Section 3.2) and one verification. Research is ongoing into alternative forms for the perturbations that could reduce this time.

Table 3 gives the parameter sets for a range of security levels, corresponding to k-bit security, and the performance (in terms of signatures and verifications per second) for each of the recommended parameter sets. We compare signature times to a single ECC point multiplication with precomputation from [4]; without precomputation the number of ECC signatures/second goes down by a factor of 3.5-4. We compare verification times to ECDSA verification times without memory constraints from [4]. As in Tables 1 and 2, NTRUSign times given are for the entire scheme (including hashing, etc), not just the primitive operation, while ECDSA times are for the primitive operation alone.

Above the 80-bit security level, NTRUSign signatures are smaller than the corresponding RSA signatures. They are larger than the corresponding ECDSA signatures by a factor of about 4. An NTRUSign private key consists of sufficient space to store f and g for the private key, plus sufficient space to store f_i, g_i and h_i for each of the B perturbation bases. Each f and g can be stored in $2N$ bits, and each h can be stored in $N \log_2(q)$ bits, so the total storage requred for the one-perturbation case is is $16N$ bits for the 80- to 128-bit parameter sets below and $17N$ bits for the 160- to 256-bit parameter sets, or approximately twice the size of the public key.

Parameters				public key and signature size				sign/s			vfy/s		
k	N	d	q	NTRU	ECDSA key	ECDSA sig	RSA	NTRU	ECDSA	Ratio	NTRU	ECDSA	Ratio
80	157	29	256	1256	192	384	1024	4560	5140	0.89	15955	1349	11.83
112	197	28	256	1576	224	448	~ 2048	3466	3327	1.04	10133	883	11.48
128	223	32	256	1784	256	512	3072	2691	2093	1.28	7908	547	14.46
160	263	45	512	2367	320	640	4096	1722	—	—	5686	—	—
192	313	50	512	2817	384	768	7680	1276	752	1.69	4014	170	23.61
256	349	75	512	3141	512	1024	15360	833	436	1.91	3229	100	32.29

Table 3. Performance measures for different NTRUSign parameter sets

6 Security: overview

We quantify security in terms of bit strength k, evaluating how much effort an attacker has to put in to break a scheme. All the attacks we consider here have variable running times, so we describe the strength of a parameter set using the notion of *cost*. For an algorithm $\mathcal{A}$ with running time t and probability of success ε, the cost is defined as

$$C_{\mathcal{A}} = t/\varepsilon \ .$$

This definition of cost is not the only one that could be used. For example, in the case of indistinguishability against adaptive chosen-ciphertext attack the attacker outputs a single bit $i \in \{0, 1\}$, and obviously has a chance of success of at least $\frac{1}{2}$. Here the probability of success is less important than the attacker's *advantage*, defined as

$$\mathrm{adv}(\mathcal{A}(\mathsf{ind})) = 2.(\Pr[\mathsf{Succ}[\mathcal{A}]] - 1/2) \ .$$

However, in this paper the cost-based measure of security is appropriate.

Our notion of cost is derived from [25] and related work. An alternate notion of cost, which is the definition above multiplied by the amount of memory used, is proposed in [38]. The use of this measure would allow significantly more efficient parameter sets, as the meet-in-the-middle attack described in Section 7.1 is essentially a time-memory tradeoff that keeps the product of time and memory constant. However, current practice is to use the measure of cost above.

We also acknowledge that the notion of comparing public-key security levels with symmetric security levels, or of reducing security to a single headline measure, is inherently problematic — see an attempt to do so in [33], and useful comments on this in [22]. In particular, extrapolation of breaking times is an inexact science, the behavior of breaking algorithms at high security levels is by definition untested, and one can never disprove the existence of an algorithm that attacks NTRUEncrypt (or any other system) more efficiently than the best currently known method.

7 Common security considerations

This section deals with security considerations that are common to NTRUEncrypt and NTRUSign.

Most public key cryptosystems, such as RSA [35] or ECC [24, 30], are based on a one-way function for which there is one best-known method of attack: factoring in the case of RSA, Pollard-rho in the case of

ECC. In the case of NTRU, there are *two* primary methods of approaching the one-way function, both of which must be considered when selecting a parameter set.

7.1 Combinatorial Security

Polynomials are drawn from a known space $\mathcal{S}$. This space can best be searched by using a combinatorial technique originally due to Odlyzko [17], which can be used to recover f or g from h or r and m from e. We denote the combinatorial security of polynomials drawn from $\mathcal{S}$ by Comb$[\mathcal{S}]$

$$\text{Comb}[\mathcal{B}_N(d)] \geq \frac{\binom{N/2}{d/2}}{\sqrt{N}} \ . \tag{10}$$

For trinary polynomials in $\mathcal{T}_N(d)$, we find

$$\text{Comb}[\mathcal{T}(d)] > \binom{N}{d+1}/\sqrt{N}. \tag{11}$$

For product-form polynomials in $\mathcal{P}_N(d)$, defined as polynomials of the form $\mathsf{a} = \mathsf{a}_1 * \mathsf{a}_2 + \mathsf{a}_3$, where $\mathsf{a}_1, \mathsf{a}_2, \mathsf{a}_3$ are all binary with $d_{a_1}, d_{a_2}, d_{a_3}$ 1s respectively, $d_{a1} = d_{a2} = d_{a3} = d_a$, and there are no further constraints on a, we find [21]:

$$\begin{aligned}\text{Comb}[\mathcal{P}_N(d)] \geq \min\Bigg(&\binom{N - \lceil N/d \rceil}{d-1}^2, \\ &\max\left(\binom{N - \lceil \frac{N}{d} \rceil}{d-1}\binom{N - \lceil \frac{N}{d-)} \rceil}{d-2}, \binom{N}{2d}\right), \\ &\max\left(\binom{N}{d}\binom{N}{d-1}, \binom{N - \lceil \frac{N}{2d} \rceil}{2d-1}\right) \\ &\Bigg)\end{aligned}$$

7.2 Lattice Security

An ntru public key h describes a $2N$-dimensional NTRU lattice containing the private key (f, g) or (f, F). When f is of the form $\mathsf{f} = 1 + \mathsf{pF}$, the best lattice attack on the private key involves solving a Close Vector Problem (CVP).[1] When f is not of the form $\mathsf{f} = 1 + \mathsf{pF}$, the best lattice attack involves solving an Approximate Shortest Vector Problem (apprSVP). Experimentally, it has been found that an NTRU lattice of this form can usefully be characterized by two quantities

$$\begin{aligned} a &= N/q, \\ c &= \sqrt{4\pi e \|\mathsf{F}\| \|\mathsf{g}\| / q} \quad (\mathsf{NTRUEncrypt}), \\ &= \sqrt{4\pi e \|\mathsf{f}\| \|\mathsf{F}\| / q} \quad (\mathsf{NTRUSign}). \end{aligned}$$

(For product-form keys the norm $\|\mathsf{F}\|$ is variable but always obeys $|\mathsf{F}| \geq \sqrt{D(N-D)/N}$, $D = d^2 + d$. We use this value in calculating the lattice security of product-form keys, knowing that in practice the value of c will typically be higher.)

This is to say that for constant (a, c), the experimentally observed running times for lattice reduction behave roughly as

$$\log(T) = AN + B \ ,$$

for some experimentally-determined constants A and B.

Table 4 summarizes experimental results for breaking times for NTRU lattices with different (a, c) values. We represent the security by the constants A and B. The breaking time in terms of bit security is $AN + B$. It may be converted to time in MIPS-years using the equality 80 bits $\sim 10^{12}$ MIPS-years.

[1] Coppersmith and Shamir [6] propose related approaches which turn out not to materially affect security.

c	a	A	B
1.73	0.53	0.3563	−2.263
2.6	0.8	0.4245	−3.440
3.7	2.7	0.4512	+0.218
5.3	1.4	0.6492	−5.436

Table 4. Extrapolated bit security constants depending on (c, a).

For constant (a, c), increasing N increases the breaking time exponentially. For constant (a, N), increasing c increases the breaking time. For constant (c, N), increasing a decreases the breaking time, although the effect is slight. More details on this table are given in [15].

Note that the effect of moving from the "standard" NTRUEncrypt lattice to the "transpose" NTRUSign lattice is to increase c by a factor of $(N/12)^{1/4}$. This allows for a given level of lattice security at lower dimensions for the transpose lattice than for the standard lattice. Since NTRUEncrypt uses the standard lattice, NTRUEncrypt key sizes given in [21] are greater than the equivalent NTRUSign key sizes at the same level of security.

The technique known as *zero-forcing* [15, 28] can be used to reduce the dimension of an NTRU lattice problem. The precise amount of the expected performance gain is heavily dependent on the details of the parameter set; we refer the reader to [15, 28] for more details. In practice this reduces security by about 6-10 bits.

7.3 Other Security Considerations

The following parameter selection criteria must also be taken into account for both NTRUEncrypt and NTRUSign.

Choosing N — The degree parameter N must be prime. (See [7].)

8 NTRUEncrypt security considerations

Parameter sets for NTRUEncrypt at a k-bit security level are selected subject to the following constraints:

- The work to recover the private key or the message through lattice reduction must be at least k bits, where bits are converted to MIPS-years using the equality 80 bits $\sim 10^{12}$ MIPS-years.
- The work to recover the private key or the message through combinatorial search must be at least 2^k binary convolutions.
- There must be no decryption failures.

8.1 Decryption Failure Security

NTRU decryption can fail on validly encrypted messages if the `center` method returns the wrong value of A, or if the coefficients of $\mathsf{prg} + \mathsf{fm}$ do not lie in an interval of width q. Decryption failures leak information about the decrypter's private key [16, 34]. The recommended parameter sets ensure that decryption failures will not happen by setting q to be greater than the maximum possible width of $\mathsf{prg} + \mathsf{m} + \mathsf{pFm}$. q should be as small as possible while respecting this bound, as lowering q increases the lattice constant c and hence the lattice security. Centering then becomes simply a matter of reducing into the interval $[0, q - 1]$.

It would be possible to improve performance by relaxing the final condition to require only that the probability of a decryption failure was less than 2^{-K}. However, this would require improved techniques for estimating decryption failure probabilities.

8.2 N, q and p

The small and large moduli p and q must be relatively prime in the ring $\mathcal{R}$. Equivalently, the three quantities

$$p, \quad q, \quad X^N - 1$$

must generate the unit ideal in the ring $\mathbb{Z}[X]$. (As an example of why this is necessary, in the extreme case that p divides q, the plaintext is equal to the ciphertext reduced modulo p.)

8.3 Factorization of $X^N - 1 \pmod q$

If $\mathsf{F}(X)$ is a factor of $X^N - 1 \pmod q$, and if $\mathsf{h}(X)$ is a multiple of $\mathsf{F}(X)$, i.e., if $\mathsf{h}(X)$ is zero in the field $K = (\mathbb{Z}/q\mathbb{Z})[X]/\mathsf{F}(X)$, then an attacker can recover the value of $\mathsf{m}(X)$ in the field K.

If q has order $t \pmod N$, then

$$X^N - 1 \equiv (X-1)\mathsf{F}_1(X)\mathsf{F}_2(X)\cdots\mathsf{F}_{(N-1)/t}(X) \quad \text{in } (\mathbb{Z}/q\mathbb{Z})[X] \ ,$$

where each $\mathsf{F}_i(X)$ has degree t and is irreducible mod q. If $\mathsf{F}_i(X)$ has degree t, the probability that $\mathsf{h}(X)$ or $\mathsf{r}(X)$ is divisible by $\mathsf{F}_i(X)$ is presumably $1/q^t$. To avoid attacks based on the factorization of h or r, we will require that for each prime divisor P of q, the order of $P \pmod N$ must be $N-1$ or $(N-1)/2$. This requirement has the useful side-effect of increasing the probability that randomly chosen f will be invertible in $\mathcal{R}_q$ [36].

8.4 Information leakage from encrypted messages

The transformation $\mathsf{a} \to \mathsf{a}(1)$ is a ring homomorphism, and so the ciphertext e has the property that

$$\mathsf{e}(1) = \mathsf{r}(1)\mathsf{h}(1) + \mathsf{m}(1) \ .$$

An attacker will know $\mathsf{h}(1)$, and for many choices of parameter set $\mathsf{r}(1)$ will also be known. Therefore, the attacker can calculate $\mathsf{m}(1)$. The larger $|\mathsf{m}(1) - N/2|$ is, the easier it is to mount a combinatorial or lattice attack to recover the msssage, so the sender should always ensure that $\|\mathsf{m}\|$ is sufficiently large. In these parameter sets, we set a value d_{m_0} such that there is a probability of less than 2^{-40} that the number of 1s or 0s in a randomly generated m is less than d_{m_0}. We then calculate the security of the ciphertext against lattice and combinatorial attacks in the case where m has exactly this many 1s and require this to be greater than 2^k for k bits of security.

8.5 NTRUEncrypt security: summary

In this section we present a summary of the security measures for the parameter sets under consideration. Table 5 gives security measures for binary keys. Table 6 gives security measures for product-form keys. The measure b_{latt} is the lattice security in bits without taking zero-forcing into account, r is the number of zeroes forced in zero-forcing, and $b_{\text{latt}}^{\text{zf}}$ is the measure including zero-forcing. In the product-form parameters table, $\mathsf{Hw}(\mathsf{F})$ is the Hamming weight of F. d_{m_0} is the same in both the binary and product-form case and is omitted from the product-form table.

9 NTRUSign security considerations

This section considers security considerations that are specific to NTRUSign.

k	N	d	d_{m0}	q	$c(\mathsf{f},\mathsf{g})$	$c(\mathsf{r},\mathsf{m})$	b_{latt}	r	$b_{\text{latt}}^{\text{zf}}$
80	251	48	70	197	2.93	2.77	103.1	29	97.98
112	347	66	108	269	2.94	2.83	143.9	31	138.26
128	397	74	128	307	2.93	2.84	165.1	33	159.17
160	491	91	167	367	2.98	2.90	205.0	35	198.75
192	587	108	208	439	2.97	2.91	245.7	37	239.21
256	787	140	294	587	2.95	2.91	330.6	41	323.45

Table 5. NTRUEncrypt security measures for different values of k using binary polynomials.

k	N	d	$\mathsf{Hw}(\mathsf{F})$	q	$c(\mathsf{f},\mathsf{g})$	$c(\mathsf{r},\mathsf{m})$	b_{latt}	r	$b_{\text{latt}}^{\text{zf}}$
80	251	8	72	293	2.57	2.43	87.2	20	80.1
112	347	11	132	541	2.21	2.13	117.8	16	118.7
128	397	12	156	659	2.24	2.17	136.3	17	136.6
160	491	15	210	967	2.08	2.02	171.3	16	170.1
192	587	17	306	1229	2.02	1.97	203.3	14	204.6
256	787	22	462	2027	1.78	1.75	278.1	14	276.7

Table 6. NTRUEncrypt security measures for different values of k using product-form polynomials

9.1 Security against forgery

We quantify the probability that an adversary, without knowledge of f, g, can compute a signature s on a given document D. The constants $N, q, \delta, \beta, \mathcal{N}$ must be chosen to ensure that this probability is less than 2^{-k}, where k is the desired bit level of security. To investigate this some additional notation will be useful:

1. Expected length of s: $\mathcal{E}_s$
2. Expected length of $t - m$: $\mathcal{E}_t$

By $\mathcal{E}_s$, $\mathcal{E}_t$ we mean respectively the expected values of $\|s\|$ and $\|t - m\|$ (appropriately reduced mod$q$) when generated by the signing procedure described in Section 3.2. These will be independent of m but dependent on N, q, δ. A genuine signature will then have expected length

$$\mathcal{E} = \sqrt{\mathcal{E}_s^2 + \beta^2 \mathcal{E}_t^2}$$

and we will set

$$\mathcal{N} = \rho\sqrt{\mathcal{E}_s^2 + \beta^2 \mathcal{E}_t^2}. \tag{12}$$

As in the case of recovering the private key, an attack can be made by combinatorial means, by lattice reduction methods or by some mixing of the two. By balancing these approaches we will determine the optimal choice of β, the public scaling factor for the second coordinate.

9.2 Combinatorial forgery

Let us suppose that $N, q, \delta, \beta, \mathcal{N}, h$ are fixed. An adversary is given m, the image of a digital document D under the hash function H. His problem is to locate an s such that

$$\|(s \bmod q, \beta(h * s - m) \bmod q)\| < \mathcal{N}.$$

In particular, this means that for an appropriate choice of $k_1, k_2 \in R$

$$(\|(s + k_1 q\|^2 + \beta^2 \|h * s - m + k_2 q)\|^2)^{1/2} < \mathcal{N}.$$

A purely combinatorial attack that the adversary can take is to choose s at random to be quite small, and then to hope that the point $h*s-m$ lies inside of a sphere of radius $\mathcal{N}/\beta$ about the origin after its coordinates are reduced mod$q$. The attacker can also attempt to combine guesses. Here, the attacker would calculate a series of random s_i and the corresponding t_i and $t_i - m$, and file the t_i and the $t_i - m$ for future reference. If

a future s_j produces a t_j that is sufficiently close to $t_i - m$, then $(s_i + s_j)$ will be a valid signature on m. As with the previous meet-in-the-middle attack, the core insight is that filing the t_i and looking for collisions allows us to check l^2 t-values while generating only l s-values.

An important element in the running time of attacks of this type is the time that it takes to file a t value. We are interested not in exact collisions, but in two t_i that lie close enough to allow forgery. In a sense, we are looking for a way to file the t_i in a spherical box, rather than in a cube as is the case for the similar attacks on private keys. It is not clear that this can be done efficiently. However, for safety, we will assume that the process of filing and looking up can be done in constant time, and that the running time of the algorithm is dominated by the process of searching the s-space. Under this assumption, the attacker's expected work before being able to forge a signature is:

$$p(N, q, \beta, \mathcal{N}) < \sqrt{\frac{\pi^{N/2}}{\Gamma(1 + N/2)} \cdot \left(\frac{\mathcal{N}}{q\beta}\right)^N}. \tag{13}$$

If k is the desired bit security level it will suffice to choose parameters so that the right hand side of (13) is less than 2^{-k}.

9.3 Signature forgery through lattice attacks

On the other hand the adversary can also launch a lattice attack by attempting to solve a closest vector problem. in particular, he can attempt to use lattice reduction methods to locate a point $(s, \beta t) \in L_h(\beta)$ sufficiently close to $(0, \beta m)$ that $\|(s, \beta(t - m))\| < \mathcal{N}$. We'll refer to $\|(s, \beta(t - m))\|$ as the norm of the intended forgery.

The difficulty of using lattice reduction methods to accomplish this can be tied to another important lattice constant:

$$\gamma(N, q, \beta) = \frac{\mathcal{N}}{\sigma(N, q, \delta, \beta)\sqrt{2N}}. \tag{14}$$

This is the ratio of the required norm of the intended forgery over the norm of the expected smallest vector of $L_h(\beta)$, scaled by $\sqrt{2N}$. For usual NTRUSign parameters the ratio, $\gamma(N, q, \beta)\sqrt{2N}$, will be larger than 1. Thus with high probability there will exist many points of $L_h(\beta)$ that will work as forgeries. The task of an adversary is to find one of these without the advantage that knowledge of the private key gives. As $\gamma(N, q, \beta)$ decreases and the ratio approaches 1 this becomes measurably harder.

Experiments have shown that for fixed $\gamma(N, q, \beta)$ and fixed N/q the running times for lattice reduction to find a point $(s, t) \in L_h(\beta)$ satisfying

$$\|(s, t - m)\| < \gamma(N, q, \beta)\sqrt{2N}\sigma(N, q, \delta, \beta)$$

behave roughly as

$$\log(T) = AN + B$$

as N increases. Here A is fixed when $\gamma(N, q, \beta), N/q$ are fixed, increases as $\gamma(N, q, \beta)$ decreases and increases as N/q decreases. Experimental results are summarized in Table 7.

Our analysis shows that lattice strength against forgery is maximized, for a fixed N/q, when $\gamma(N, q, \beta)$ is as small as possible. We have

$$\gamma(N, q, \beta) = \rho\sqrt{\frac{\pi e}{2N^2 q} \cdot (\mathcal{E}_s^2/\beta + \beta\mathcal{E}_t^2)} \tag{15}$$

and so clearly the value for β which minimizes γ is $\beta = \mathcal{E}_s/\mathcal{E}_t$. This optimal choice yields

$$\gamma(N, q, \beta) = \rho\sqrt{\frac{\pi e \mathcal{E}_s \mathcal{E}_t}{N^2 q}}. \tag{16}$$

Referring to (13) we see that increasing β has the effect of improving combinatorial forgery security. Thus the optimal choice will be the minimal $\beta \geq \mathcal{E}_s/\mathcal{E}_t$ such that $p(N, q, \beta, \mathcal{N})$ defined by (13) is sufficiently small.

An adversary could attempt a mixture of combinatorial and lattice techniques, fixing some coefficients and locating the others via lattice reduction. However, as explained in [19], the lattice dimension can only be reduced a small amount before a solution becomes very unlikely. Also, as the dimension is reduced, γ decreases, which sharply increases the lattice strength at a given dimension.

bound for γ and N/q	$\omega_{\text{lf}}(N)$
$\gamma < 0.1774$ and $N/q < 1.305$	$0.995113N - 82.6612$
$\gamma < 0.1413$ and $N/q < 0.707$	$1.16536N - 78.4659$
$\gamma < 0.1400$ and $N/q < 0.824$	$1.14133N - 76.9158$

Table 7. Bit security against lattice forgery attacks, ω_{lf}, based on experimental evidence for different values of $(\gamma, N/q)$

10 Transcript security

NTRUSign is not zero-knowledge. This means that, while NTRUEncrypt can have provable security (in the sense of a reduction from an online attack method to a purely offline attack method), there is no known method for establishing such a reduction with NTRUSign. NTRUSign is different in this respect from established signature schemes such as ECDSA and RSA-PSS, which have reductions from online to offline attacks. Research is ongoing into quantifying what information is leaked from a transcript of signatures and how many signatures an attacker needs to observe to recover the private key or other information that would allow the creation of forgeries. This section summarizes existing knowledge about this information leakage.

10.1 Transcript security for raw NTRUSign

First, consider raw NTRUSign. In this case, an attacker studying a long transcript of valid signatures will have a list of pairs of polynomials of the form

$$\mathsf{s} = \epsilon \mathsf{f} + \epsilon' \mathsf{g}, \quad \mathsf{t} - \mathsf{m} = \epsilon \mathsf{F} + \epsilon' \mathsf{G}$$

where the coeffcients of ϵ, ϵ' lie in the range $[-1/2, 1/2]$. In other words, the signatures lie inside a parallopiped whose sides are the good basis vectors. The attacker's challenge is to discover one edge of this parallelopiped.

Since the ϵs are random, they will average to 0. To base an attack on averaging s and $t - m$, the attacker must find something that does not average to zero. To do this he uses the *reversal* of s and $t - m$. The reversal of a polynomial a is the polynomial

$$\bar{\mathsf{a}}(X) = \mathsf{a}(X^{-1}) = \mathsf{a}_0 + \sum_{i=1}^{N-1} \mathsf{a}_{N-i} X^i.$$

We then set

$$\hat{\mathsf{a}} = \mathsf{a} * \bar{\mathsf{a}}.$$

Notice that $\hat{\mathsf{a}}$ has the form

$$\hat{\mathsf{a}} = \sum_{k=0}^{N-1} \left(\sum_{i=0}^{N-1} \mathsf{a}_i \mathsf{a}_{i+k} \right) X^k.$$

In particular, $\hat{\mathsf{a}}_0 = \sum_i \mathsf{a}^2$. This means that as the attacker averages over a transcript of $\hat{\mathsf{s}}, \hat{\mathsf{t} - \mathsf{m}}$, the cross-terms will essentially vanish and the attacker will recover

$$\langle \hat{\epsilon}_0 \rangle (\hat{\mathsf{f}} + \hat{\mathsf{g}}) = \frac{N}{12} (\hat{\mathsf{f}} + \hat{\mathsf{g}})$$

for s and similarly for $\mathsf{t} - \mathsf{m}$, where $\langle . \rangle$ denotes the average of . over the transcript.

We refer to the product of a measurable with its reverse as its *second moment*. In the case of raw NTRUSign, recovering the second moment of a transcript reveals the Gram Matrix of the private basis. Experimentally, it appears that significant information about the Gram Matrix is leaked after 10,000 signatures for all of the parameter sets in this paper. A recent paper by Nguyen and Regev [31] demonstrates an attack on parameter sets without perturbations that combines Gram matrix recovery with creative use of averaging moments over the signature transcript to recover the private key after seeing a transcript of approximately 70,000 signatures. It is not clear that this attack has been optimized and the use of unperturbed NTRUSign is strongly discouraged.

Obviously, something must be done to reduce information leakage from transcripts, and this is the role played by perturbations.

10.2 Transcript security for NTRUSign with perturbations

In the case with B perturbations, the expectation of $\hat{\mathsf{s}}$ and $\hat{\mathsf{t}} - \hat{\mathsf{m}}$ is (up to lower order terms)

$$E(\hat{\mathsf{s}}) = (N/12)(\hat{\mathsf{f}}_0 + \hat{\mathsf{g}}_0 + \ldots + \hat{\mathsf{f}}_B + \hat{\mathsf{g}}_B)$$

and

$$E(\hat{\mathsf{t}} - \hat{\mathsf{m}}) = (N/12)(\hat{\mathsf{f}}_0 + \hat{\mathsf{g}}_0 + \ldots + \hat{\mathsf{f}}_B + \hat{\mathsf{g}}_B).$$

Note that this second moment is no longer a Gram matrix but the sum of $(B+1)$ Gram matrices. Likewise, the signatures in a transcript do not lie within a parallelopiped but within the sum of $(B+1)$ parallelopipeds.

This complicates matters for an attacker. The best currently known technique for $B = 1$ is to calculate

$$\begin{aligned} \text{the second moment } & \langle \hat{\mathsf{s}} \rangle \\ \text{the fourth moment } & \langle \hat{\mathsf{s}}^2 \rangle \\ \text{the sixth moment } & \langle \hat{\mathsf{s}}^3 \rangle \ . \end{aligned}$$

Since, for example, $\langle \hat{\mathsf{s}} \rangle^2 \neq \langle \hat{\mathsf{s}}^2 \rangle$, the attacker can use linear algebra to eliminate f_1 and g_1 and recover the Gram matrix, whereupon the attack of [31] can be used to recover the private key. It is an interesting open research question to determine whether there is any method open to the attacker that enables them to eliminate the perturbation bases without recovering the sixth moment (or, in the case of B perturbation bases, the $(4B+2)$-th moment). For now, the best known attack is this algebraic attack, which requires the recovery of the sixth moment. It is an open research problem to discover analytic attacks based on signature transcripts that improve on this algebraic attack.

We now turn to estimate τ, the length of transcript necessary to recover the sixth moment. Consider an attacker who attempts to recover the sixth moment by averaging over τ signatures and rounding to the nearest integer. This will give a reasonably correct answer when the error in many coefficients (say at least half) is less than $1/2$. To compute the probability that an individual coefficient has an error less than $1/2$, write $(12/N)\hat{s}$ as a main term plus an error, where the main term converges to $\hat{f}_0 + \hat{g}_0 + \hat{f}_1 + \hat{g}_1$. The error will converge to 0 at about the same rate as the main term converges to its expected value. If the probability that a given coefficient is further than $1/2$ from its expected value is less than $1/(2N)$ then we can expect at least half of the coefficients to round to their correct values. (Note that this convergence cannot be speeded up using lattice reduction in, for example, the lattice $\hat{h}$, because the terms $\hat{f}$, $\hat{g}$ are unknown and are larger than the expected shortest vector in that lattice).

The rate of convergence of the error and its dependence on τ can be estimated by an application of Chernoff-Hoeffding techniques [26], using an assumption of a reasonable amount of independence and uniform distribution of random variables within the signature transcript. This assumption appears to be justified by experimental evidence, and in fact benefits the attacker by ensuring that the cross-terms converge to zero.

Using this technique, we estimate that to have a single coefficient in the $2k$-th moment with error less than $\frac{1}{2}$, the attacker must analyze a signature transcript of length $\tau > 2^{2k+4} d^{2k}/N$. Here d is the number of 1's in the trinary key. Experimental evidence for the second moment indicates that the required transcript length will in fact be much longer than this. For one perturbation, the attacker needs to recover the sixth moment accurately, leading to required transcript lengths $\tau > 2^{30}$ for all the recommended parameter sets in this paper.

11 NTRUSign security: summary

In this section we present a summary of the security measures for the parameter sets under consideration. The security measures have the following meanings:

ω_{lk}	The security against key recovery by lattice reduction
c	The lattice characteristic c that governs key recovery times
ω_{cmb}	The security against key recovery by combinatorial means
ω_{frg}	The security against forgery by combinatorial means
γ	The lattice characteristic γ that governs forgery times
ω_{lf}	The security against forgery by lattice reduction

The parameter sets in Table 8 were generated with $\rho = 1.1$ and selected to give the shortest possible signing time σ_S.

Parameters						Security Measures						
k	N	d	q	β	$\mathcal{N}$	ω_{cmb}	c	ω_{lk}	ω_{frg}	γ	ω_{lf}	$\log_2(\tau)$
80	157	29	256	0.38407	150.02	104.43	5.34	93.319	80	0.139	102.27	31.9
112	197	28	256	0.51492	206.91	112.71	5.55	117.71	112	0.142	113.38	31.2
128	223	32	256	0.65515	277.52	128.63	6.11	134.5	128	0.164	139.25	32.2
160	263	45	512	0.31583	276.53	169.2	5.33	161.31	160	0.108	228.02	34.9
192	313	50	512	0.40600	384.41	193.87	5.86	193.22	192	0.119	280.32	35.6
256	349	75	512	0.18543	368.62	256.48	7.37	426.19	744	0.125	328.24	38.9

Table 8. Parameters and relevant security measures for trinary keys, one perturbation, $\rho = 1.1$, q = power of 2

12 Conclusions: the NTRU algorithms after quantum computers

At the moment it is unclear what effect quantum computers may have on the security of the NTRU algorithms. The paper [27] describes a quantum algorithm that square-roots asymptotic lattice reduction running times for a specific lattice reduction algorithm. However, since in practice lattice reduction algorithms perform much better than they are theoretically predicted to, it is not clear what effect this improvement in asymptotic running times has on practical security. On the combinatorial side, Grover's algorithm [9] provides a means for square-rooting the time for a brute-force search. However, the combinatorial security of NTRU keys depends on a meet-in-the-middle attack and we are not currently aware of any quantum algorithms to speed this up.

At the moment it seems reasonable to speculate that quantum algorithms will be discovered that will square-root times for both lattice reduction and meet-in-the-middle searches. If this is the case, NTRU key sizes will have to approximately double and running times will increase by a factor of approximately 4 to give the same security levels. As demonstrated in the performance tables in this paper, this still results in performance that is competitive with public key algorithms that are in use today. As quantum computers are seen to become more and more feasible, NTRUEncrypt and NTRUSign should be seriously studied with a view to wide deployment.

References

1. ANSI X9.62, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), 1999.
2. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In Proc. of Eurocrypt '94, volume 950 of LNCS, pages 92–111. IACR, SpringerVerlag, 1995.
3. D. Boneh, Simplified OAEP for the RSA and Rabin functions, In proceedings of Crypto '2001, Lecture Notes in Computer Science, Vol. 2139, Springer-Verlag, pp. 275-291, 2001
4. M. Brown, D. Hankerson, J. López, and A. Menezes, Software Implementation of the NIST Elliptic Curves Over Prime Fields, *CT-RSA 2001*, D. Naccache (Ed.), LNCS 2020, 250–265, Springer-Verlag, 2001.
5. Consortium for Efficient Embedded Security, *Efficient Embedded Security Standard #1 version 2*, available from `http://www.ceesstandards.org`.
6. D. Coppersmith and A. Shamir, *Lattice Attack on NTRU*, Advances in Cryptology - Eurocrypt'97, Springer-Verlag
7. C. Gentry, Key recovery and message attacks on NTRU-composite, *Advances in Cryptology —Eurocrypt '01*, LNCS 2045. Springer-Verlag, 2001
8. C. Gentry, M Szydlo, *Cryptanalysis of the Revised NTRU SignatureScheme*, Advances in Cryptology—Eurocrypt '02, Lecture Notes in Computer Science, Springer-Verlag, 2002.
9. L. Grover, *A fast quantum mechanical algorithm for database search*, Proceedings, 28th Annual ACM Symposium on the Theory of Computing, 1996.
10. D. Hankerson, J. Hernandez, A. Menezes, *Software implementation of elliptic curve cryptography over binary fields*, Proceedings of CHES 2000, Lecture Notes in Computer Science, 1965 (2000), 1-24
11. J. Hoffstein, J. Pipher, J.H. Silverman, *NTRU: A new high speed public key cryptosystem*, in Algorithmic Number Theory (ANTS III), Portland, OR, June 1998, Lecture Notes in Computer Science 1423 (J.P. Buhler, ed.), Springer-Verlag, Berlin, 1998, 267–288.

12. J. Hoffstein and J. H. Silverman. Optimizations for NTRU. In Publickey Cryptography and Computational Number Theory. DeGruyter, 2000. Available at [4].
13. J. Hoffstein and J. H. Silverman, Random Small Hamming Weight Products With Applications To Cryptography, Discrete Applied Mathematics, to appear, Available from `http://www.ntru.com`.
14. J. Hoffstein and J. H. Silverman. Invertibility in truncated polynomial rings. Technical report, NTRU Cryptosystems, October 1998. Report #009, version 1, available at `http://www.ntru.com`.
15. J. Hoffstein, J. H. Silverman, W. Whyte, Estimated Breaking Times for NTRU Lattices, Technical report, NTRU Cryptosystems, June 2003 Report #012, version 2, available at `http://www.ntru.com`.
16. N. A. Howgrave-Graham, P. Nguyen, D. Pointcheval, J. Proos, J. H. Silverman, A. Singer, W. Whyte, *The Impact of Decryption Failures on the Security of NTRU Encryption*, Advances in Cryptology—Crypto 2003, Lecture Notes in Compputer Science 2729, Springer-Verlag, 2003, 226-246.
17. N. A. Howgrave-Graham, J. H. Silverman, W. Whyte, A Meet-in-the-Middle Attack on an NTRU Private key, Technical report, NTRU Cryptosystems, June 2003. Report #004, version 2, available at `http://www.ntru.com`.
18. N. Howgrave-Graham, J. H. Silverman, A. Singer and W. Whyte. NAEP: Provable Security in the Presence of Decryption Failures IACR ePrint Archive, Report 2003-172, `http://eprint.iacr.org/2003/172/`
19. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. Silverman, W. Whyte, NTRUSign: Digital Signatures Using the NTRU Lattice, CT-RSA 2003,
20. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. Silverman, W. Whyte, Performance Improvements and a Baseline Parameter Generation Algorithm for NTRUSign, Workshop on Mathematical Problems and Techniques in Cryptology, Barcelona, Spain, June 2005
21. N. A. Howgrave-Graham, J. H. Silverman, W. Whyte, Choosing Parameter Sets for NTRUEncrypt with NAEP and SVES-3, CT-RSA 2005, to appear.
22. B. Kaliski, Comments on SP 800-57, Recommendation for Key Management, Part 1: General Guidelines. Available from `http://csrc.nist.gov/CryptoToolkit/kms/CommentsSP800-57Part1.pdf`.
23. E. Kiltz, J. Malone-Lee, *A General Construction of IND-CCA2 Secure Public Key Encryption*, In: Cryptography and Coding, pages 152–166. Springer-Verlag, December 2003.
24. N. Koblitz. *Elliptic curve cryptosystems*. Mathematics of Computation, 48, pages 203–209, 1987.
25. A. K. Lenstra, E. R. Verheul, *Selecting cryptographic key sizes*, Journal of Cryptology vol. 14, no. 4, 2001, 255-293. Available from `http://www.cryptosavvy.com`.
26. Kirill Levchenko, Chernoff Bound, available at `http://www.cs.ucsd.edu/ klevchen/techniques/chernoff.pdf`
27. C. Ludwig: A Faster Lattice Reduction Method Using Quantum Search, TU-Darmstadt Cryptography and Computeralgebra Technical Report No. TI-3/03, revised version published in Proc. of ISAAC 2003
28. A. May, J.H. Silverman, *Dimension reduction methods for convolution modular lattices*, in Cryptography and Lattices Conference (CaLC 2001), J.H. Silverman (ed.), Lecture Notes in Computer Science 2146, Springer-Verlag, 2001
29. T. Meskanen and A. Renvall. Wrap Error Attack Against NTRUEncrypt. Proc. of WCC '03.
30. V. Miller. *Uses of elliptic curves in cryptography*. In Advances in Cryptology: Crypto '85, pages 417–426, 1985.
31. P. Nguyen, O. Regev *Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures* Eurocrypt 2006.
32. NIST, Digital Signature Standard, FIPS Publication 186-2, February 2000.
33. NIST Special Publication 800-57, Recommendation for Key Management, Part 1: General Guideline, January 2003. Available from `http://csrc.nist.gov/CryptoToolkit/kms/guideline-1-Jan03.pdf`.
34. J. Proos *Imperfect Decryption and an Attack on the NTRU Encryption Scheme*, IACR ePrint Archive, report 02/2003. `http://eprint.iacr.org/2003/002/`.
35. R. Rivest, A. Shamir, L. M. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, Communications of the ACM, 21 (1978), 120-126.
36. J. H. Silverman, Invertibility in Truncated Polynomial Rings, Technical report, NTRU Cryptosystems, October 1998 Report #009, version 1, available at `http://www.ntru.com`.
37. P. Shor, Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer. Preliminary version appeared in Proc. of 35th Annual Symp. on Foundations of Computer Science, Santa Fe, NM, Nov 20-22, 1994. Final version published in SIAM J. Computing 26 (1997) 1484. Published in SIAM J.Sci.Statist.Comput.26:1484,1997 e-Print Archive: quant-ph/9508027
38. Robert D. Silverman, A Cost-Based Security Analysis of Symmetric and Asymmetric Key Lengths. *RSA Labs Bulletin 13*, April 2000. available from `http://www.rsasecurity.com/rsalabs`.

Implementation of Improved "Quantum Public-Key Cryptosystem"

Toshiyuki Miyazawa, Tetsutaro Kobayashi, Satoshi Oda,
Ichizo Nakamura, and Atsushi Kanai

NTT Information Sharing Platform Laboratories
1-1 Hikari-no-oka, Yokosuka-shi, Kanagawa-ken, 239-0847 Japan
Email: miyazawa.toshiyuki@lab.ntt.co.jp
Tel: +81-468-59-2193
Fax: +81-468-59-3365

Abstract. In 2000, Okamoto, Tanaka, and Uchiyama proposed the concept of the "quantum public-key cryptosystem" and a concrete scheme called OTU2000 [15]. OTU2000 is secure against attacks using quantum computers and uses these computers to generate a key pair. Thus, it is thought that OTU2000 will be actualized after quantum computers are actualized. We improve the original OTU2000 scheme so that it can be executed using current computers and show implementation results, using the multiplicative group of the ring of integers modulo a 640-bit composite number. The key generation procedure requires less than 2 hours, the encryption procedure requires approximately 4.4 msec, and the decryption procedure requires approximately 7.6 msec on a PC (Pentium 4 3.2 GHz). These results show that the OTU2000 can be implemented using current computers.

1 Introduction

1.1 Background

In 2000, Okamoto, Tanaka, and Uchiyama proposed the concept of the "quantum public-key cryptosystem" [15]. The quantum public-key cryptosystem is a secure public-key cryptosystem that deters attacks using quantum computers. In [15], the authors also proposed the concrete scheme (OTU2000) that satisfies the requirements for the quantum public-key cryptosystem. This scheme is a knapsack-based cryptosystem that overcomes the weak points of previous schemes, e.g., the Chor-Rivest scheme [4] and Merkle-Hellman scheme [10]. The subset sum problem, upon which knapsack-based cryptosystems are based, is an NP-complete problem. Since no efficient quantum algorithm solving NP-complete problems is known, OTU2000 is secure against attacks using quantum computers.

OTU2000 also uses quantum computers to generate a key pair. The key generation procedure of this scheme includes solving the discrete logarithm problem (DLP) in a finite field, and Shor's algorithm [23] can solve the DLP using quantum computers. Thus, it is thought that the OTU2000 will be actualized after quantum computers are actualized.

1.2 Related Work

The DLP over a small finite field can be solved using classical computers. Recently, techniques to solve the DLP have improved. The fastest method to solve the DLP

is the number field sieve method [20]. Currently, the world record for solving the DLP is held by Joux and Lercier, and they solved the DLP over a 130-digit prime finite field using the number field sieve method [7]. This requires approximately 3 weeks. Another technique for solving the DLP is Pollard's method [17] and the baby-step/giant-step method [18], which requires $O(\sqrt{q})$ time, where q is the order of the cyclic group.

We use these methods to generate the key for OTU2000 instead of using quantum computers.

1.3 Our Results

We improve the original OTU2000 scheme so that it can be executed using current computers and without quantum computers. In [15], quantum computers are employed only to solve the DLP over a finite field and the other computations are executable by current computers. The proposed concept is to employ the ring of integers modulo a composite number, instead of the finite field.

We also implemented the proposed scheme, using N as a composite number of 640 bits and $N = q_1q_2q_3q_4q_5$, where q_i is a random prime of 128 bits. As a result, the key generation procedure requires less than 2 hours, the encryption procedure requires approximately 4.4 msec, and the decryption procedure requires approximately 7.6 msec on a Pentium 4 3.2-GHz PC. This result shows that our scheme is executable on current computers.

Roadmap This paper is organized as follows. We improve the original scheme of OTU2000 so that it can be easily executed on current computers. The improved scheme is described in Section 2. Next, we discuss the security of the improved scheme in Section 3. In Section 4, we present the selection of parameters and the execution data.

2 Improved Version of OTU2000

Our improved scheme is based on the rational version of OTU2000 in Appendix A of [15]. In the original scheme the key generation algorithm must solve the DLP in a finite field. Thus, quantum computers are used to generate a key pair.

Here, we improve the original scheme by replacing the finite field with the ring of integers modulo a composite number. Details are given below:

Key Generation

Input : Positive integers n,h
Output : Secret key $SK = (g, d, N, p_1, ..., p_n, k)$ and public key $PK = (n, k, b_1, ..., b_n)$

Step 1. Randomly select h primes $q_1, ..., q_h$, that satisfy the following conditions:
  (a) The length of q_j is $\frac{n}{h}$ bits for $j = 1, ..., h$.
  (b) The greatest common number of $q_{j_1} - 1$ and $q_{j_2} - 1$ is 2 for $1 \leq j_1 < j_2 \leq h$.

Step 2. Compute $N = \prod_{j=1}^{h} q_j$.

Step 3. Compute $L = 2^{-h+1} \prod_{j=1}^{h} (q_j - 1)$, that is the largest common multiple of $\{q_j - 1 \ : \ j = 1, ..., h\}$.

Step 4. Randomly select generator $g_j \in (\mathbb{Z}/q_j\mathbb{Z})^\times$ for $j = 1, ..., h$.

Step 5. Compute $g \in (\mathbb{Z}/N\mathbb{Z})^\times$ such that $g \equiv g_j \pmod{q_j}$ for any $1 \leq j \leq h$ using the Chinese remainder theorem.

Step 6. Choose n integers $p_1, ..., p_n$, that satisfy the following conditions:
  – $p_1, ..., p_n$ are co-prime.
  – $\left(\frac{p_i}{q_1} \right) = ... = \left(\frac{p_i}{q_h} \right)$ for $i = 1, ..., n$.

Step 7. Compute discrete logarithms $a_{i,j}$ such that

$$p_i \equiv g^{a_{i,j}} \pmod{q_j},$$

where $i = 1, ..., n$ and $j = 1, ..., h$.

Step 8. Compute $a_i \in \mathbb{Z}/L\mathbb{Z}$ for $i = 1, ..., n$ such that

$$a_i \equiv a_{i,j} \pmod{q_j - 1}$$

for $j = 1, ..., h$ (by using a variant of the Chinese remainder theorem).

Step 9. Randomly select $d \in \mathbb{Z}/L\mathbb{Z}$.

Step 10. Compute $b_i = a_i + d \mod L$ for each $1 \leq i \leq n$.

Step 11. Compute integer k such that for any subset $\{p_{i_1}, ..., p_{i_k}\} \subset \{p_1, ..., p_n\}$,

$$\prod_{s=1}^{k} p_{i_s} < N.$$

Step 12. Output secret key $SK = (g, d, N, p_1, ..., p_n, k)$ and public key $PK = (n, k, b_1, ..., b_n)$.

We note that the choice of p_i is restricted. Since N is a composite number, the multiplicative group $(\mathbb{Z}/N\mathbb{Z})^\times$ is not cyclic. Hence, it is not true that $p_i \in \langle g \rangle$ for any $p_i \in (\mathbb{Z}/N\mathbb{Z})^\times$. To overcome this problem, we select modulus N whose factor satisfies the condition in Step 1 and the generator $g \in (\mathbb{Z}/N\mathbb{Z})^\times$ in Step 4. If such an N and g are employed, the following proposition can be proved:

Proposition 1. *Let $N = q_1 \cdots q_h$, $g \in (\mathbb{Z}/N\mathbb{Z})^\times$ and $g_j \in (\mathbb{Z}/q_j\mathbb{Z})^\times$ $(1 \leq j \leq h)$ be as above. Then the natural homomorphism*

$$f : \langle g \rangle \longrightarrow \prod_{j=1}^{h} (\mathbb{Z}/q_j\mathbb{Z})^\times$$

$$f(g^x) = (g^x \bmod q_1, \ g^x \bmod q_2, \ ..., \ g^x \bmod q_h)$$

is injective and

$$\mathrm{Im}(f) = \{(g_j^{\alpha_j} \bmod q_j)_{1\le j\le h}\ ;\ \alpha_{j_1} \equiv \alpha_{j_2} \pmod 2 \ \textit{ for any } 1 \le j_1 < j_2 \le h\}.$$

Proof. The injectivity of f is trivial from Chinese remainder theorem. Let $S = \{(g_j^{\alpha_j} \bmod q_j)_{1\le j\le h}\ ;\ \alpha_{j_1} \equiv \alpha_{j_2} \pmod 2 \text{ for any } 1 \le j_1 < j_2 \le h\}$. Since $S \supset \mathrm{Im}(f)$, we will show $S \subset \mathrm{Im}(f)$.

Let $(g_j^{\alpha_j} \bmod q_j)_{1\le j\le h} \in S$. From the condition of q_j's, $q_j - 1$ is equal to $2r_j$ for some $r_j \in \mathbb{Z}$ such that $\gcd(r_{j_1}, r_{j_2}) = 1$ for any $1 \le j_1 < j_2 \le h$. Then $x \in \mathbb{Z}$ can be found such that it satisfies the congruent equations

$$\begin{aligned} x &\equiv \alpha_1 \pmod 2 \\ x &\equiv \alpha_1 \pmod{r_1} \\ x &\equiv \alpha_2 \pmod{r_2} \\ &\vdots \\ x &\equiv \alpha_h \pmod{r_h} \end{aligned}$$

using the Chinese remainder theorem. We can see that $f(g^x) = (g_j^{\alpha_j} \bmod q_j)_{1\le j\le h}$; hence, $S \subset \mathrm{Im}(f)$. □

Because of Proposition 1, $f : \langle g\rangle \to S$ is an isomorphism. Then, instead of checking whether $p_i \in \langle g\rangle$, we can check the following condition:

$$a_{j_1} \equiv a_{j_2} \pmod 2 \ (1 \le j_1 < j_2 \le h), \ \text{ where } p_i \equiv g^{a_{j_t}} \pmod{q_{j_t}}.$$

This condition is checked in Step 6 of the key generation algorithm.

The encryption and decryption algorithms are the same as the original version in OTU2000.

Encryption

Input : Public key PK and plaintext M with $\lfloor \log_2 \binom{n}{k} \rfloor$ bits.
Output : Ciphertext C

Step 1. Encode M into binary string $m = (m_1, ..., m_n)$ of length n and Hamming weight k (i.e., having exactly k 1's) as follows:
  (a) Set $l \leftarrow k$
  (b) For i from 1 to n perform the following:
    If $M \ge \binom{n-i}{l}$ then set $m_i \leftarrow 1$, $M \leftarrow M - \binom{n-i}{l}$, $l \leftarrow l - 1$.
    Otherwise set $m_i \leftarrow 0$.(Note that $\binom{l}{0} = 1$ for $l \ge 0$, and $\binom{0}{l} = 0$ for $l \ge 1$.)
Step 2. Compute ciphertext C as $C = \sum_{i=1}^{n} m_i b_i$

Decryption

Input : Secret key SK and ciphertext C
Output : Plaintext M

Step 1. Compute $r = C - kd$.
Step 2. Compute $u = g^r \pmod N$.
Step 3. For i from 1 to n perform the following:
If $p_i|u$, then set $m_i \leftarrow 1$, otherwise set $m_i \leftarrow 0$.
Step 4. Set $m = (m_1, ..., m_n)$.
Step 5. Decode m into plaintext M as follows:
(a) Set $M \leftarrow 0,\ l \leftarrow k$.
(b) For i from 1 to n perform the following:
If $m_i = 1$, then set $M \leftarrow M + \binom{n-i}{l}$ and $l \leftarrow l - 1$.

Correctness and remarks

1. [**Decryption**] We show that the decryption works. We observe that

$$\begin{aligned} u &\equiv g^r \pmod N \\ &\equiv g^{C-kd} \pmod N \\ &\equiv g^{(\sum_{i=1}^n m_i b_i)-kd} \pmod N \\ &\equiv g^{\sum_{i=1}^n m_i a_i} \pmod N \\ &\equiv \prod_{i=1}^n (g^{a_i})^{m_i} \pmod N \\ &\equiv \prod_{i=1}^n p_i^{m_i} \pmod N \\ &= \prod_{i=1}^n p_i^{m_i}. \end{aligned}$$

Since the product of any k elements among $\{p_1, \cdots, p_n\}$ is less than N and $p_1, \cdots, p_n$ are co-prime, $\prod_{i=1}^n p_i^{m_i} \pmod N$ is uniquely factorized by $\{p_1, \cdots, p_n\}$. Thus, a ciphertext is uniquely deciphered to m (i.e., M).
2. [**Density**] Since $b_i \in [0, L]$ and $L \simeq 2^{-h+1}N$, the bit length of b_i is slightly less than n. Hence, density [8] $d = \frac{n}{\log_2 \max b_i}$ of our scheme is slightly greater than one.

3 Security Consideration

In this section, we discuss the security parameters for the proposed scheme.

3.1 Solving Knapsack

Since the security of OTU2000 depends on the subset sum problem, we should use parameters such that the knapsack becomes sufficiently difficult to solve.

Meet-in-the-Middle Attacks The obvious meet-in-the-middle attack computes two lists of $\binom{n}{k/2}$: namely L_1 the list of all $\sum_{i=1}^{n} b_i x_i$ where $x = (x_1, ..., x_n)$ has the Hamming weight of $k/2$, and L_2 the list of all $c - \sum_{i=1}^{n} b_i x_i$ where $x = (x_1, ..., x_n)$ has the Hamming weight of $k/2$. Then the attack determines the intersection of L_1 and L_2, since collisions between L_1 and L_2 clearly disclose the knapsack solutions. The cost is approximately $\binom{n}{k/2}$.

There is however a more efficient meet-in-the-middle attack. This attack is performed by adapting Coppersmith's meet-in-the-middle attack for solving a discrete log with a low Hamming weight exponent (see [19]). Roughly speaking, we apply a $O(\sqrt{k})$ meet-in-the-middle attack at the cost of $\binom{n/2}{k/2}$.

1. Select a random subset $\mathcal{B} \subset \{1, ..., n\}$ of size $n/2$.
2. Let L_1 be the list of all $\sum_{i=1}^{n} b_i x_i$ where $x = (x_1, ..., x_n) \in \mathcal{B}$ has the Hamming weight of $k/2$, and L_2 be the list of all $c - \sum_{i=1}^{n} b_i x_i$ where $x = (x_1, ..., x_n) \in \mathcal{B}$ has the Hamming weight of $k/2$.
3. Find collisions between L_1 and L_2.
4. If there are no collisions, restart at the beginning.

In [19], it was shown that the number of iterations to find a collision is $O(\sqrt{k})$. This is related to the probability of exactly splitting the k-element subset defining the solution of the knapsack problem when selecting $\mathcal{B}$. The overall cost (up to some logarithmic factor) is $O(\sqrt{k})\binom{n/2}{k/2}$. Thus, parameters n and k must be selected in such a way that $\binom{n}{k/2}$ and $\sqrt{k}\binom{n/2}{k/2}$ are both sufficiently large.

Lattice Attack Recently Nguyễn and Stern showed efficient provable reductions from the knapsack problem with a low Hamming weight to the lattice problems: the shortest vector problem (SVP) and the closest vector problem (CVP) [13]. These studies indicate that OTU2000 can be attacked by the CVP/SVP-oracle.

However, the "ideal" CVP/SVP-oracle does not exist that can solve the SVP/CVP in a polynomial time because these problems are NP-hard [1, 2]. There exist only lattice reduction algorithms, for example LLL [9] and BKZ [21], that perform as an approximate SVP/CVP-oracle. If a dimension of the lattice is sufficiently large, these algorithms cannot solve exactly SVP/CVP in practice.

Today, the world record for the largest lattice dimension solved is the 350-dimensional GGH decryption challenge [11]. In the case of a lattice constructed from a knapsack-type cryptosystem, Schnorr and Hörner failed to decrypt the Chor-Rivest ciphertext that corresponds to a lattice dimension of around 200-250 [22]. The smallest parameter for the NTRU cryptosystem, whose security is based on lattice problems, corresponds to a 502-dimensional lattice. According to these results, in order to avoid a lattice attack using reduction algorithms, the proposed scheme is sufficiently secure if the lattice dimension becomes 500 or more.

Moreover, if the density is close to one, a lattice attack using reduction algorithms is more difficult in practice [22]. The density of our scheme is slightly greater than one. Thus, a lattice attack against our scheme using reduction algorithms cannot be easy.

3.2 Key Exposure Attack

Nguyễn and Stern showed that if N is composite and one of q_j can be guessed, there is a possibility that the whole secret key can be recovered [12]. Here, we show the outline of a possible attack.

1. **Recovering p_i's**
   For any tuple (p_i, p_j, p_l),
   $$p_i^{b_l-b_j} p_j^{b_i-b_j} \equiv p_l^{b_i-b_j} p_j^{b_l-b_j} \pmod{q_j}.$$
   Because $p_j \lesssim N^{1/k}$, a tuple (p_i, p_j, p_l) can be found by exhaustive search. Once such a tuple (p_i, p_j, p_l) has been found, other p_i's can be found easily.
2. **Recovering N**
   If all p_i are known, N can be found using the following procedure:
   (a) Compute $(\varepsilon_2, ..., \varepsilon_n) \in \mathbb{Z}^{n-1}$ satisfying the linear equation
   $$\sum_{i=2}^{m} \varepsilon_i (b_i - b_1) = 0$$
   using a lattice reduction algorithm.
   (b) Let A and $B \subset \{2, ..., n\}$ be $A = \{i : 2 \le i \le n \ \& \ \varepsilon_i > 0\}$ and $B = \{i : 2 \le i \le n \ \& \ \varepsilon_i < 0\}$.
   (c) Since
   $$\prod_{i \in B} p_i^{-\varepsilon_i} p_1^{\sum_{i \in B} -\varepsilon_i} \equiv \prod_{i \in A} p_i^{\varepsilon_i} p_1^{\sum_{i \in A} \varepsilon_i} \pmod{N},$$
   subtracting the left side of the congruence from the right side, the multiple of N can be found.
   (d) Given two such $(\varepsilon_2, ..., \varepsilon_n)$, N can be found using the gcd computation.
3. **Recovering g**
   If N and two of the p_i's are known, g can be found using the following procedure:
   (a) Factor $N = q_1 \cdots q_h$. [1]
   (b) Compute $g_j \in (\mathbb{Z}/q_j\mathbb{Z})^\times$ such that
   $$p_{i_1} p_{i_2}^{-1} \equiv g_j^{b_{i_1} - b_{i_2}} \pmod{q_j}$$
   for each j $(1 \le j \le h)$.
   (c) Compute $g \in (\mathbb{Z}/N\mathbb{Z})^\times$ such that
   $$g \equiv g_j \pmod{q_j} \text{ for all } j \ (1 \le j \le h)$$
   using the Chinese reminder theorem.
4. **Recovering d**
   If N, g, and one of p_i's are known, g^d can be found from $g^d \equiv g^{b_i}/p_i \pmod{N}$. For decryption, d is not necessary because instead of computing $u \equiv g^{C-kd} \pmod{N}$, $u \equiv g^C/(g^d)^k \pmod{N}$ can be computed, and the rest of the decryption process can be applied.

According to the above consideration, if one of q_j is guessed, the proposed scheme can be solved. Thus, all q_j should be sufficiently large.

[1] It would be an appropriate assumption in the "post-quantum" age.

4 Implemented Data

In this section, we show the performance of the improved scheme.

4.1 Parameter Selection

We set $n = 640$ and $h = 5$, i.e., $|q_j| = 128$. To maximize the Hamming weight, k, we select p_i such that p_i are the smallest n primes that satisfy the condition in Step 6 of the key generation algorithm. Then, k is equal to 38. The computational cost of the meet-in-the-middle attack is $\binom{n/2}{k} \simeq 2^{164.38}$ and $\sqrt{k}\binom{n/2}{k/2} \simeq 2^{103.15}$. The lattice dimension of this parameter is nearly equal to 640 (see [22, 13]), i.e., greater than 500. Thus, $n = 640$ is considered to be secure against a lattice attack.

The number of 128 bit primes is approximately equal to $2^{83.87}$. Then $|q_j| = 128$ is considered to be secure against a key exposure attack.

The Bit Length of Each Parameter In the case of $n = 640$, $h = 5$, and $k = 38$, the length of each parameter is given in Table 1.

parameters	length (bits)
Plaintext	204
Cipertext	641
Public Key	407,056
Secret Key	12,802

Table 1. Bit Length of Each Parameter ($n = 640, k = 38, h = 5$)

4.2 Performance of Our Scheme

Using Smooth Numbers In order to generate a key pair more easily, we select q_j such that $q_j - 1$ is 2^{40}-smooth for each $j = 1, ..., 5$. We implemented the proposed scheme in the environment described below.

- Language ... GP/PARI Calculator Version 2.2.9 [16]
- OS ... Windows XP
- CPU ... Pentium 4, 3.2 GHz
- Memory ... 2 GB

The GP/PARI calculator uses the baby-step/giant-step method to solve the DLP.

The average cost and best/worst cost of each procedure is shown in Table 2. In our trial, we randomly selected $N = q_1q_2q_3q_4q_5$ satisfying the condition in Step1 of the key generation algorithm. We tried the key generation 10 times, and the encryption and the decryption 10,000 times.

	Average	Best	Worst
Key Generation	116 minutes	67 minutes	280 minutes
Encryption	4.4 ms	–	–
Decryption	7.6 ms	–	–

Table 2. Performance of Proposed Scheme ($n = 640, h = 5$)

Using Number Field Sieve The fastest method for solving the DLP is the number field sieve [20]. The current world record for solving the DLP in a 130-digit prime field is held by Joux and Lercier using the number field sieve method [7]. It takes about 3 weeks.

To generate a key pair, we need only calculate the DLP in a 128-bit prime. Even if $q_j - 1$ is not smooth, it easy enough to solve the DLP for the proposed key generation algorithm.

Comparison with RSA and elliptic curve cryptosystem We compare the performance of our scheme with that of RSA and ECC. We implemented the primitives of the RSA and ElGamal elliptic curve cryptosystem (EC-ElGamal) in the environment described above. The average of each procedure is shown in Table 3.

In our trial, we tried the encryption and the decryption 10,000 times. We use the RSA of a 1024 bit modulus and the exponent $e = 65537$ of a public key. The parameters of the EC-ElGamal are `secp160k1` of a 160 bit prime field in [24].

	Proposed Scheme	1024-bit RSA	160-bit EC-ElGamal
Encryption	4.4 ms	0.3 ms	5.7 ms
Decryption	7.6 ms	27.1 ms	2.9 ms

Table 3. Performance Comparison of Proposed Scheme, RSA, and ECC

The performances of encryption and decryption of proposed scheme is not different from RSA and ECC.

5 Conversions to Stronger Security

In the previous sections, we proposed a trapdoor *one-way* function, or a primitive functionality of a public-key encryption. It is easy to convert practically such a primitive public-key function to a public-key encryption scheme with the strongest security (non-malleable against adaptively chosen ciphertext attacks) in the random oracle model. For example, some generic and efficient conversions have been proposed, e.g., Fujisaki-Okamoto [6].

6 Concluding Remarks

This paper presented a new approach to achieve a public-key cryptosystem based on a knapsack (subset-sum) problem that is secure even after the quantum computer is actualized. This new approach employed the ring of integers modulo a composite number $\mathbb{Z}/N\mathbb{Z}$ that provides us with a huge degree of freedom for choosing trapdoor parameters and enables key generation without quantum computers. We implemented the proposed scheme and showed that the key-generation procedure requires less than 2 hours, the encryption procedure requires approximately 4.4 msec, and the decryption procedure requires approximately 7.6 msec on a common PC. These results show that our scheme can be executed on current computers.

Finally, we note that our approach can be applied to the number field case of OTU2000.

Acknowledgement

The authors thank Dr. Phong Q. Nguyễn and Dr. Jacques Stern for their helpful comments concerning the security of our scheme. We also thank Dr. Tatsuaki Okamoto for suggesting the improved scheme.

References

1. M. Ajtai. "The shortest vector problem in L_2 is NP-hard for randomized reductions," in Proc. of 30th STOC. ACM (1998).
2. P. van Emde Boas. "Another NP-complete problem and the complexity of computing short vectors in a lattice," Technical report, Mathematische Instituut, University of Amsterdam, Report 81-04 (1981).
3. M. Bellare and P. Rogaway,"Optimal Asymmetric Encryption," Proc. of Eurocrypt'94, LNCS 950, Springer-Verlag pp. 92-111 (1995).
4. B. Chor and R.L. Rivest "A knapsack-type public key cryptosystem based on arithmetic in finite fields," IEEE Trans. on Information Theory-34, pp. 901-909 (1988)
5. M.J. Coster, A. Joux, B.A. LaMacchia, A.M. Odlyzko, C.P. Schnorr, and J. Stern. "Improved low-density subset sum algorithms," Computational Complexity, 2, pp. 111-128 (1992)
6. E. Fujisaki and T. Okamoto,"Secure Integration of Asymmetric and Symmetric Encryption Schemes," Proc. of Crypto'99, Springer-Verlag, LNCS 1666, pp. 535–554 (1999).
7. A. Joux and R. Lercier. "`Discrete logarithms in GF(p) --- 130 digits`," 18 Jun 2005. NMBRTHRY Mailing List. Available at `http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0506&L=nmbrthry&F=&S=&P=2037`
8. J.C. Lagarias and A.M. Odlyzko. "Solving low-density subset sum problems," Journal of the Association for Computing Machinery, 32, pp. 229-246 (1985)
9. A.K. Lenstra, H.W. Lenstra, and L. Lovász. "Factoring polynomials with rational coefficients," Mathematische Annalen 261, 515-534 (1982)
10. R.C. Merkle and M. Hellman. "New directions in cryptography," IEEE Trans. on Information Theory-19, pp. 525-530 (1973)
11. P.Q. Nguyễn, "Cryptoanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97," Proc. of Asiacrypt '99, LNCS 1666, pp. 288-304, Springer-Verlag (1999).
12. P.Q. Nguyễn and J. Stern, private communication (2005-2006).
13. P.Q. Nguyễn and J. Stern "Adapting Density Attacks to Low-Weight Knapsacks," Proc. of Asiacrypt 2005, LNCS 3788, pp. 41-58, Springer-Verlag (2005).
14. T.Okamoto and D.Pointcheval, "REACT : Rapid Enhanced-security Asymmetric Cryptosystem Transform", In CT-RSA'01, LNCS 2020, pp.159-175, Springer-Verlag,Berlin (2001).
15. T. Okamoto, K. Tanaka, and S. Uchiyama."Quantum Public Key Cryptosystems," Proc. of CRYTPTO 2000, LNCS 1880, pp. 147-165, Springer-Verlag (2000).
16. PARI/GP, version `2.1.7`, Bordeaux, 2005, `http://pari.math.u-bordeaux.fr/`.

17. J. Pollard. "Monte Carlo methods for index computation for index computations mod p," Math. Comp., 32, pp. 918-924 (1978).
18. D. Shanks. "Class number, a theory of factorization and genera," In Proc. Symp. Pure Math. 20(1971), AMS, Providence, R.I., pp. 415-440, 1971.
19. D.R. Stinson. "Some baby-step giant-step algorithms for the low Hamming weight discrete logarithm problem," Mathematics of Computation, 71 (2002), pp. 379-391.
20. O. Schirokauer. "Discrete logarithms and local units, " Phil. Trans. R. Soc. Lond. A 345 (1993).
21. C.P. Schnorr and M. Euchner, "Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems," Mathematical Programming, Vol. 66, pp. 181-191 (1994)
22. C. Schnorr and H. Hörner, "Attacking the Chor-Rivest Cryptosystem by Improved Lattice Reduction," Advances in Cryptology - Eurocrypt '95 LNCS, Vol. 921, pp. 1-12, (1995)
23. P.W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," Proc. 35nd Annual Symposium on Foundations of Computer Science (Shafi Goldwasser, ed.), IEEE Computer Society Press (1994), 124-134.
24. Standards for Efficient Cryptography Group, "SEC 2: Recommended Elliptic Curve Domain Parameters", `http://www.secg.org/download/aid-386/sec2_final.pdf` (2000)

Post-quantum code-based cryptography

Nicolas Sendrier

Institut National de Recherche en Informatique et en Automatique

Equivalent Keys in $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Public Key Systems

Christopher Wolf[1,2], and Bart Preneel[1]

[1]K.U.Leuven, ESAT-COSIC
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
{Christopher.Wolf, Bart.Preneel}@esat.kuleuven.be
or chris@Christopher-Wolf.de
http://www.esat.kuleuven.ac.be/cosic/

[2]École Normale Supérieure, Département d'Informatique
45 rue d'Ulm, F-75230 Paris Cedex 05, France
Christopher.Wolf@ens.fr or chris@Christopher-Wolf.de

Abstract

$\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key schemes have been suggested back in 1985 by Matsumoto and Imai as an alternative for the RSA scheme. Since then, several other schemes have been proposed, for example Hidden Field Equations, Unbalanced Oil and Vinegar schemes, and Stepwise Triangular Schemes. All these schemes have a rather large key space for a secure choice of parameters. Surprisingly, the question of equivalent keys has not been discussed in the open literature until recently. In this article, we show that for all basic classes mentioned above, it is possible to reduce the private — and hence the public — key space by several orders of magnitude. For the Matsumoto-Imai scheme, we are even able to show that the reductions we found are the only ones possible, *i.e.*, that these reductions are tight. While the theorems developed in this article are of independent interest themselves as they broaden our understanding of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key systems, we see applications of our results both in cryptanalysis and in memory efficient implementations of $\mathcal{MQ}$-schemes.

Keywords: $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Polynomials, Public Key signature, Hidden Field Equations, Matsumoto-Imai scheme A, C*, Unbalanced Oil and Vinegar, Stepwise Triangular Systems

1 Initial Considerations

In the last 20 years, several schemes based on the problem of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations (or $\mathcal{MQ}$ for short) have been proposed. The most important ones certainly are MIA / C* [MI88] and Hidden Field Equations (HFE, [Pat96b]) plus their variations MIA- / C^{*--}, HFE-, HFEv, and HFEv- [KPG99, Pat96a, Pat96b]. Both classes have been used to construct signature schemes for the European cryptography project NESSIE [NES], namely the MIA- variation in Sflash [CGP03], the HFEv- variation in Quartz [CGP01] and the HFE- variation in the tweaked version Quartz-7m [WP04]. Unbalanced Oil and Vinegar schemes [KPG99] and Stepwise Triangular Schemes [WBP04] are also important in practice. While the first is secure with the correct choice of parameters, the second forms the basis of nested constructions like the enhanced TTM [YC04], Tractable Rational Maps [WHL$^+$05], or Rainbow [DS05].

The aim of this paper is to systematically study the question of equivalent keys of $\mathcal{MQ}$-schemes. At first glance, this question seems to be purely theoretical. But for practical applications, we need memory and time efficient instances of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key systems. One important point in this context is the overall *size* of the private key: in restricted environments such as smart cards, we want it as small as possible. Hence, if we can show that a given private key is only a representative of a much larger class of equivalent private keys, it makes sense to compute (and store) only a normal form of this key. Similar, we should construct new $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic schemes such that they do not have a large number of equivalent private keys but only a small number, preferable only one, per equivalence class. This way, we make optimal use of the randomness in the private key space and neither waste computation time nor storage space without any security benefit.

All systems based on $\mathcal{MQ}$-equations use a public key of the form

$$p_i(x_1, \ldots, x_n) := \sum_{1 \le j \le k \le n} \gamma_{i,j,k} x_j x_k + \sum_{j=1}^{n} \beta_{i,j} x_j + \alpha_i \,,$$

with $n \in \mathbb{Z}^+$ variables and $m \in \mathbb{Z}^+$ equations. Moreover, we have $1 \leq i \leq m; 1 \leq j \leq k \leq n$ and $\alpha_i, \beta_{i,j}, \gamma_{i,j,k} \in \mathbb{F}$ (constant, linear, and quadratic terms). We write the set of all such systems of polynomials as $\mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$. Moreover, the private key consists of the triple $(S, \mathcal{P}', T)$ where $S \in \text{Aff}^{-1}(\mathbb{F}^n), T \in \text{Aff}^{-1}(\mathbb{F}^m)$ are bijective affine transformations. Details on affine transformation are given in Section 2.1). Moreover, we have $\mathcal{P}' \in \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$ is a polynomial-vector $\mathcal{P}' := (p'_1, \ldots, p'_m)$ with m components; each component is a polynomial in n variables $x'_1, \ldots, x'_n$. Throughout this paper, we will denote components of this private vector $\mathcal{P}'$ by a prime $'$. In contrast to the public polynomial vector $\mathcal{P} \in \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$, the private polynomial vector $\mathcal{P}'$ does allow an efficient computation of $x'_1, \ldots, x'_n$ for given $y'_1, \ldots, y'_m$. Still, the goal of $\mathcal{MQ}$-schemes is that this inversion should be hard if the public key $\mathcal{P}$ alone is given. The main difference between $\mathcal{MQ}$-schemes lies in their special construction of the central equations $\mathcal{P}'$ and consequently the trapdoor they embed into a specific class of $\mathcal{MQ}$-problems. An introduction to $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key systems is given in [WP05c].

1.1 Related Work

In their cryptanalysis of HFE, Kipnis and Shamir report the existence of "isomorphic keys" [KS99]. A similar observation for Unbalanced Oil and Vinegar Schemes can be found in [KPG99]. In both cases, there has not been a systematic study of the structure of equivalent key classes. In addition, Patarin observed the existence of some equivalent keys for MIA / C* [Pat96a] — however, his method is different from the one presented in this article, as he concentrated on modifying the central monomial rather than using special affine transformations. Moreover, Toli observed that there exists an additive sustainer in the case of Hidden Field Equations [Tol03] but did not extend his result to other $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic schemes. Additive sustainers will be introduced in Section 3.1. In the case of symmetric ciphers, [BCBP03] used a similar idea in the study of S-boxes. A different angle of the idea of equivalent keys can be found in [HWyCL05] where the authors compute normal forms of the *public* key. Main reason here is to save some memory in the public but particularily in the private key. Using the techniques suggested in [HWyCL05], the latter can be reduced by up to 50%.

This article is based on the two conference papers [WP05b, WP05a] which deal with the classes MIA, HFE, and UOV. In this article, the proofs have been simplified and also extended to the STS class. In addition, a tightness proof for the case of MIA is given.

1.2 Outline

This paper is organized as follows: after this general introduction, we move on to the necessary mathematical background in Section 2. This includes particularly a definition of the term *equivalent keys*. In Section 3, we concentrate on a subclass of affine transformations, denoted *sustaining transformations*, which can be used to generate equivalent keys. These transformations are applied to different variations of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations in Section 4. In Section 5, we give a tightness proof for the case of MIA/MIO. This paper concludes with Section 6.

2 Mathematical Considerations

Before discussing concrete schemes, we start with some general observations and definitions. Obviously, the most important term in this article is "equivalent private keys". We give a graphical representation of this idea in Figure 1. We can also express this idea in the following definition:

Definition 2.1 *We call two private keys*

$$(S, \mathcal{P}', T), (\tilde{S}, \tilde{\mathcal{P}}', \tilde{T}) \in \mathit{Aff}^{-1}(\mathbb{F}^n) \times \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m) \times \mathit{Aff}^{-1}(\mathbb{F}^m)$$

"equivalent" if they lead to the same public key, i.e., if we have

$$T \circ \mathcal{P}' \circ S = \mathcal{P} = \tilde{T} \circ \tilde{\mathcal{P}}' \circ \tilde{S}\,.$$

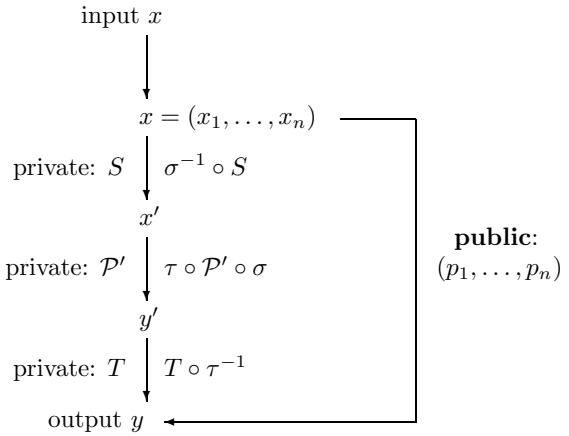


Fig. 1: Equivalent private keys using affine transformations σ, τ

In the above definition, $\mathrm{Aff}^{-1}(\cdot)$ denotes the class of bijective affine transformations. We give more details on affine transformations in Section 2.1. In order to find equivalent keys, we consider the following transformations:

DEFINITION 2.2 *Let $(S, \mathcal{P}', T) \in \mathit{Aff}^{-1}(\mathbb{F}^n) \times \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m) \times \mathit{Aff}^{-1}(\mathbb{F}^m)$, and consider the four transformations $\sigma, \sigma^{-1} \in \mathit{Aff}^{-1}(\mathbb{F}^n)$ and $\tau, \tau^{-1} \in \mathit{Aff}^{-1}(\mathbb{F}^m)$. Moreover, let*

$$\mathcal{P} = T \circ \tau^{-1} \circ \tau \circ \mathcal{P}' \circ \sigma \circ \sigma^{-1} \circ S\,. \tag{1}$$

We call the pair $(\sigma, \tau) \in \mathit{Aff}^{-1}(\mathbb{F}^n) \times \mathit{Aff}^{-1}(\mathbb{F}^m)$ "sustaining transformations" for an $\mathcal{MQ}$-system if the "shape" of $\mathcal{P}'$ is invariant under the transformations σ and τ. For short, we write $(\sigma, \tau) \bullet (S, \mathcal{P}', T)$ for (2.2) and (σ, τ) sustaining transformations. This idea has already been outlined in Figure 1.

Remark. In the above definition, the meaning of "shape" is still open. In fact, its meaning has to be defined for each $\mathcal{MQ}$-system individually. For example, in HFE (cf Section 4.1), it is the bounding degree $d \in \mathbb{Z}^+$ of the polynomial $P'(X') \in \mathbb{E}[X']$. In the case of MIA, the "shape" is the fact that we have a single monomial with factor 1 as the central equation (cf Section 4.2). In general and for σ, τ sustaining transformations, we are now able to produce equivalent keys for a given private key by $(\sigma, \tau) \bullet (S, \mathcal{P}', T)$. A trivial example of sustaining transformations is the identity transformation, *i.e.*, to set $\sigma = \tau = id$.

Lemma 2.3 *Let $\sigma \in \mathit{Aff}^{-1}(\mathbb{F}^n), \tau \in \mathit{Aff}^{-1}(\mathbb{F}^m)$ be sustaining transformations. If the two structures $G := (\sigma, \circ)$ and $H := (\tau, \circ)$ form a subgroup of the affine transformations, they produce equivalence relations within the private key space.*

PROOF. We start with a proof of this statement for $G := (\sigma, \circ)$. First, we have reflexivity as the identity transformation is contained in the subgroup G. Second, we also have symmetry as subgroups are closed under inversion. Third, we have transitivity as subgroups are closed under composition. Therefore, the subgroup G partitions the private key space into equivalence classes. The proof for the subgroup $H := (\tau, \circ)$ is analogous. □

Remark. We want to point out that the above proof does not use special properties of sustaining transformations, but the fact that we dealt with subgroups of the group of affine transformations. Hence, the proof does not depend on the term "shape" and is therefore valid even if the latter is not rigorously defined yet. In any case, instead of proving that sustaining transformations form a subgroup of the affine transformations, we can also consider normal forms of private keys. As we see below, normal forms have some advantages to avoid double counts in the private key space.

2.1 Affine Transformations

Given that our main tool to construct equivalent keys are special subclasses of affine transformations, we start with some general observations on them. As we only deal with bijective affine transformations $\mathrm{Aff}^{-1}(\cdot)$ and bijective linear transformations $\mathrm{Hom}^{-1}(\cdot)$ in this article, the following lemma proves useful:

Lemma 2.4 *Let $\mathbb{F}$ be a finite field with $q := |\mathbb{F}|$ elements. Then we have $\prod_{i=0}^{n-1} q^n - q^i$ invertible $(n \times n)$-matrices over $\mathbb{F}$.*

Next, we recall some basic properties of affine transformations over the finite fields $\mathbb{F}$ and $\mathbb{E}$.

Definition 2.5 *Let $M_S \in \mathbb{F}^{n \times n}$ be an invertible $(n \times n)$ matrix and $v_s \in \mathbb{F}^n$ a vector and let $S(x) := M_S x + v_s$. We call this the "matrix representation" of the affine transformation S.*

Definition 2.6 *Moreover, let $s_1, \ldots, s_n$ be n polynomials of degree 1 at most over $\mathbb{F}$, i.e., $s_i(x_1, \ldots, x_n) := \beta_{i,1}x_1 + \ldots + \beta_{i,n}x_n + \alpha_i$ with $1 \leq i, j \leq n$ and $\alpha_i, \beta_{i,j} \in \mathbb{F}$. Let $S(x) := (s_1(x), \ldots, s_n(x))$ for $x := (x_1, \ldots, x_n)$ as a vector over $\mathbb{F}^n$. We call this the "multivariate representation" of the affine transformation S.*

Remark. The multivariate and the matrix representation of an affine transformation S are interchangeable. We only need to set the corresponding coefficients to the same values: $(M_S)_{i,j} \leftrightarrow \beta_{i,j}$ and $(v_S)_i \leftrightarrow \alpha_i$ for $1 \leq i, j \leq n$. However, the first is useful in the context of matrix equations while the latter is preferable when dealing with affine transformations in the context of term substitution.

In addition, we can also use the "univariate representation" over the extension field $\mathbb{E}$ of the transformation S.

Definition 2.7 *Let $0 \leq i < n$ and $A, B_i \in \mathbb{E}$. Moreover, let the polynomial $S(X) := \sum_{i=0}^{n-1} B_i X^{q^i} + A$ be an affine transformation. We call this the "univariate representation" of the affine transformation $S(X)$.*

Lemma 2.8 *An affine transformation in univariate representation can be transfered efficiently in multivariate representation and vice versa.*

Proof. This lemma follows from [KS99, Lemmata 3.1 and 3.2] by a simple extension from the linear to the affine case. A more elaborated proof can be found in [Wol05, Lemma 2.2.7]. □

3 Sustaining Transformations

In this section, we discuss several examples of sustaining transformations. In particular, we consider their effect on the central transformation $\mathcal{P}'$.

3.1 Additive Sustainer

For $n = m$, *i.e.*, the number of equations is equal to the number of variables, let $\sigma(X) := (X + A)$ and $\tau(X) := (X + A')$ for some elements $A, A' \in \mathbb{E}$. As long as the transformations σ, τ keep the shape of the central equations $\mathcal{P}'$ invariant, they form sustaining transformations.

In particular, we are able to change the constant parts $v_s, v_t \in \mathbb{F}^n$ or $V_S, V_T \in \mathbb{E}$ of the two affine transformations $S, T \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$ to zero, *i.e.*, to obtain a new key $(\hat{S}, \hat{\mathcal{P}}', \hat{T})$ with $\hat{S}, \hat{T} \in \mathrm{Hom}^{-1}(\mathbb{F}^n)$. The constant terms of S, T have now been moved to the central equation $\mathcal{P}'$ and as a result, $\hat{S}, \hat{T}$ are now linear rather than affine transformations over $\mathbb{F}^n$.

Remark. This result is very useful for cryptanalysis as it allows us to "collect" the constant terms in the central equations $\mathcal{P}'$. For cryptanalytic purposes, we therefore only need to consider the case of linear transformations $S, T \in \mathrm{Hom}^{-1}(\mathbb{F}^n)$.

The additive sustainer also works if we interpret it over the vector space $\mathbb{F}^n$ rather than the extension field $\mathbb{E}$. To distinguish this case from the setting above, we write $a \in \mathbb{F}^n, a' \in \mathbb{F}^m$ here. In particular, we can also handle the case $n \neq m$ now. However, in this case it may happen that we have $a' \in \mathbb{F}^m$ and consequently $\tau : \mathbb{F}^m \to \mathbb{F}^m$. Nevertheless, we can still collect all constant terms in the central equations $\mathcal{P}'$.

If we look at the central equations as multivariate polynomials, the additive sustainer will affect the constants α_i and $\beta_{i,j} \in \mathbb{F}$ for $1 \leq i \leq m$ and $1 \leq j \leq n$. A similar observation is true for central equations over the extension field $\mathbb{E}$: in this case, the additive sustainer affects the additive constant $A \in \mathbb{E}$ and the linear factors $B_i \in \mathbb{E}$ for $0 \leq i < n$.

3.2 Big Sustainer

We now consider multiplication in the (big) extension field $\mathbb{E}$, *i.e.*, we have $\sigma(X) := (BX)$ and $\tau(X) := (B'X)$ for $B, B' \in \mathbb{E}^*$. Again, we obtain a sustaining transformation if this operation does not modify the shape of the central equations as $(BX), (B'X) \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$.

The big sustainer is useful if we consider schemes defined over extension fields as it does not affect the overall degree of the central equations over this extension field. Note that we only allow non-zero elements of the extension field $\mathbb{E}$ for B, B' as $BX, B'X$ are not invertible otherwise.

3.3 Small Sustainer

We now consider vector-matrix multiplication over the (small) ground field $\mathbb{F}$, *i.e.*, we have $\sigma(x) := \mathit{Diag}(b_1, \ldots, b_n)x$ and $\tau(x) := \mathit{Diag}(b'_1, \ldots, b'_m)x$ for the non-zero coefficients $b_1, \ldots, b_n, b'_1, \ldots, b'_m \in \mathbb{F}^*$ and $\mathit{Diag}(b), \mathit{Diag}(b')$ the diagonal matrices on both vectors $b \in \mathbb{F}^n$ and $b' \in \mathbb{F}^m$, respectively.

In contrast to the big sustainer, the small sustainer is useful if we consider schemes which define the central equations over the ground field $\mathbb{F}$ as it only introduces a scalar factor in the polynomials $(p'_1, \ldots, p'_m)$. As for the big sustainer, we require non-zero elements, *i.e.*, we have $b_i, b'_i \in \mathbb{F}^*$.

3.4 Permutation Sustainer

For the transformation σ, this sustainer permutes input-variables of the central equations while for the transformation τ, it permutes the polynomials of the central equations themselves. As each permutation has a corresponding, invertible permutation-matrix, both $\sigma \in S_n$ and $\tau \in S_m$ are also affine transformations. The effect of the central equations is limited to a permutation of these equations and their input variables, respectively.

3.5 Gauss Sustainer

Here, we consider Gauss operations on matrices, *i.e.*, row and column permutations, multiplication of rows and columns by scalars from the ground field $\mathbb{F}$, and the addition of two rows/columns. As all these operations can be performed by invertible matrices, they form a subgroup of the affine transformations and are hence a candidate for a sustaining transformation.

The effect of the Gauss sustainer is similar to the permutation sustainer and the small sustainer. In addition, it allows the addition of multivariate quadratic polynomials. This will not affect the shape of some $\mathcal{MQ}$-schemes.

3.6 Frobenius Sustainer

Definition 3.1 *Let $\mathbb{F}$ be a finite field with $q := |\mathbb{F}|$ elements and $\mathbb{E}$ its n-dimensional extension. Moreover, let $H := \{i \in \mathbb{Z} : 0 \leq i < n\}$. For $a, b \in H$ we call $\sigma(X) := X^{q^a}$ and $\tau(X) := X^{q^b}$* Frobenius transformations.

Obviously, Frobenius transformations are linear transformations with respect to the ground field $\mathbb{F}$. The following lemma establishes that they also form a group:

Lemma 3.2 *Frobenius transformations are a subgroup in $\mathit{Hom}^{-1}(\mathbb{F}^n)$.*

Proof. First, Frobenius transformations are linear transformations, so associativity is inherited from them. Second, the set H from Definition 3.1 is not empty for any given $\mathbb{F}$ and $n \in \mathbb{Z}^+$. Hence, the corresponding set of Frobenius transformations is not empty either. In particular, we notice that the

Frobenius transformation X^{q^0} coincides with the neutral element of the group of linear transformations $(\mathrm{Hom}^{-1}(\mathbb{F}^n), \circ)$.

In addition, the inverse of a Frobenius transformation is also a Frobenius transformation: Let $\sigma(X) := X^{q^a}$ for some $a \in H$. Working in the multiplicative group $\mathbb{E}^*$ we observe that we need $q^a \cdot A' \equiv 1 \pmod{q^n - 1}$ for $A' \in \mathbb{Z}^+$ to obtain the inverse function of σ. We notice that $A' := q^{a'}$ for $a' := n - a \pmod{n}$ yields the required and moreover $\sigma^{-1} := X^{q^{a'}}$ is a Frobenius transformation as $a' \in H$.

So all left to show is that for any given Frobenius transformations σ, τ, the composition $\sigma \circ \tau$ is also a Frobenius transformation, *i.e.*, that we have closure.

Let $\sigma(X) := X^{q^a}$ and $\tau(X) := X^{q^b}$ for some $a, b \in H$. So we can write $\sigma(X) \circ \tau(X) = X^{q^{a+b}}$. If $a + b < n$ we are done. Otherwise $n \leq a + b < 2n$, so we can write $q^{a+b} = q^{n+s}$ for some $s \in H$. Again, working in the multiplicative group E^* yields $q^{n+s} \equiv q^s \pmod{q^n - 1}$ and hence, we established that $\sigma \circ \tau$ is also a Frobenius transformation. This completes the proof that all Frobenius transformations form a group. □

Frobenius transformations usually change the degree of the central equation $\mathcal{P}'$. But taking $\tau := \sigma^{-1}$ cancels this effect and hence preserves the degree of $\mathcal{P}'$. Therefore, we can speak of a Frobenius sustainer (σ, τ). Fore a given extension field $\mathbb{E}$, there are n Frobenius sustainers.

It is tempting to extend this result to the case of powers of the characteristic of $\mathbb{F}$. However, this is not possible as $x^{\mathrm{char}\mathbb{F}}$ is not a linear transformation in $\mathbb{F}$ for $q \neq p$ where p denotes the characteristic of the finite field $\mathbb{F}$ and $q := |\mathbb{F}|$ the number of its elements.

Remark. All six sustainers presented so far form groups and hence partition the private key space into equivalence classes. The relation between partitions and groups has been previously discussed in Lemma 2.3.

3.7 Reduction Sustainer

Reduction sustainers are quite different from the transformations studied so far, because they are applied with a different construction of the trapdoor of $\mathcal{P}$. In this new construction, we define the public key equations as $\mathcal{P} := R \circ T \circ \mathcal{P}' \circ S$ where $R : \mathbb{F}^n \to \mathbb{F}^{n-r}$ denotes a *reduction* or *projection* while $S, \mathcal{P}', T$ have the same meaning as before, *i.e.*, they are affine invertible transformations and a system of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomials, respectively. Less loosely speaking, we consider the function $R(x_1, \ldots, x_n) := (x_1, \ldots, x_{n-r})$, *i.e.*, we neglect the last r components of the vector $(x_1, \ldots, x_n)$. Although this modification looks rather easy, it proves powerful to defeat a wide class of cryptographic attacks against several $\mathcal{MQ}$-schemes, including HFE and MIA, *e.g.*, the attack introduced in [FJ03].

For the corresponding sustainer, we consider the affine transformation T in matrix representation, *i.e.*, we have $T(x) := Mx + v$ for some invertible matrix $M \in \mathbb{F}^{m \times m}$ and a vector $v \in \mathbb{F}^m$. We observe that any change in the last r columns of M or v does not affect the result of R (and hence $\mathcal{P}$). Therefore, we can choose these last r columns without affecting the public key. Inspecting Lemma 2.4, we see that this gives us a total of

$$q^r \prod_{i=n-r-1}^{n-1} \left(q^n - q^i\right)$$

choices for v and M, respectively, that do not affect the public key equations $\mathcal{P}$.

When applying the reduction sustainer together with other sustainers, we have to make sure that we do not count the same transformation twice. We will show how to deal with this difficulty in the corresponding proofs.

4 Application to $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Schemes

Having all necessary tools at hand, we show now how to apply suitable sustaining transformations to the $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic schemes. We want to stress that the reductions in size we achieve in this section represent lower rather than upper bounds: additional sustaining transformations may further reduce the key space of these schemes. The only exception for this rule are the MIA/MIO class: due to the tightness

proof in Section 5, we know that only the big sustainer and the Frobenius sustainer can be applied here. Unfortunately, the details of this tightness proof are cumbersome and we do not see how it can be extended to the other schemes discussed in this section.

4.1 Hidden Field Equations

We start with the HFE class as the overall proof ideas can be demonstrated most clearly here. In fact, we will use some of these ideas again for the MIA class. The Hidden Field Equations (HFE) have been proposed by Patarin [Pat96b]. Its main characteristic is the exceptional low degree of the central polynomial $P'(X') \in \mathbb{E}[X']$.

DEFINITION 4.1 *Let $\mathbb{E}$ be a finite field and $P'(X')$ a polynomial over $\mathbb{E}$. For*

$$P'(X') := \sum_{\substack{0\le i,j\le d \\ q^i+q^j\le d}} C'_{i,j} X'^{q^i+q^j} + \sum_{\substack{0\le k\le d \\ q^k\le d}} B'_k X'^{q^k} + A'$$

$$\text{where} \begin{cases} C'_{i,j} X^{q^i+q^j} & \text{for } C'_{i,j} \in \mathbb{E} \text{ are the quadratic terms,} \\ B'_k X^{q^k} & \text{for } B'_k \in \mathbb{E} \text{ are the linear terms, and} \\ A' & \text{for } A' \in \mathbb{E} \text{ is the constant term} \end{cases}$$

and a degree $d \in \mathbb{Z}^+$, we say the central equations $\mathcal{P}'$ are in HFE-shape.

Due to the special form of $P'(X')$, we can express it as a $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equation $\mathcal{P}'$ over $\mathbb{F}$. A proof of this fact for the case $\mathbb{F} =$GF(2) can be found in [MIHM85]. It has been elaborated and further extended in [Wol05, Section 2.4]. Polynomials of cubic and higher degree have been discussed in [KS99, Lemma 3.3]. The bound of the degree of the polynomial $P'(X')$ has a different motivation: this allows efficient inversion of the equation $P(X) = Y$ for given $Y \in \mathbb{E}$ and is hence necessary to obtain efficient schemes. So the *shape* of HFE is in particular this degree d of the private polynomial P. Moreover, we observe that there are no restrictions on its coefficients $C'_{i,j}, B'_k, A' \in \mathbb{E}$ for $i, j, k \in \mathbb{Z}^+$ and $q^i, q^i + q^j \le d$. Hence, we can apply both the additive and the big sustainer from sections 3.1 and 3.2 without changing the shape of this central equation.

Theorem 4.2 *For $K := (S, P, T) \in \mathit{Aff}^{-1}(\mathbb{F}^n) \times \mathbb{E}[X'] \times \mathit{Aff}^{-1}(\mathbb{F}^n)$ a private key in HFE, we have*

$$n.q^{2n}(q^n - 1)^2$$

equivalent keys.

PROOF. To prove this theorem, we consider normal forms of private keys: let $\tilde{S} \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$ being the affine transformation we start with. First we compute $\hat{S}(X) := \tilde{S}(X) - \tilde{S}(0)$, *i.e.*, we apply the additive sustainer. Obviously, we have $\hat{S}(0) = 0$ after this transformation and hence a special fix-point. Second we define $\overline{S}(X) := \hat{S}(X).\hat{S}(1)^{-1}$, *i.e.*, we apply the big sustainer. As the transformation $\hat{S} : \mathbb{E} \to \mathbb{E}$ is a bijection and we have $\hat{S}(0) = 0$, we know that $\hat{S}(1)$ must be non-zero. Hence, we have $\overline{S}(1) = 1$, *i.e.*, we add a new fix-point but still keep the old fix-point as we have $\overline{S}(0) = \hat{S}(0) = 0$. Similar we can compute an affine transformation $\overline{T}(X)$ with $\overline{T}(0) = 0$ and $\overline{T}(1) = 1$ as a normal form of the affine transformation $\tilde{T} \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$. Note that both the additive sustainer and the big sustainer keep the degree of the central polynomial $P(X)$ so we can apply both sustainers on both sides without changing the "shape" of $P(X)$.

Applying the Frobenius sustainer is a little more technical. First we observe that this sustainer keeps the fix-points $\overline{S}(0) = \overline{T}(0) = 0$ and $\overline{S}(1) = \overline{T}(1) = 1$ so we are sure we still deal with equivalence classes, *i.e.*, each given private key has a unique normal form, even after the Frobenius sustainer has been applied. Now we pick an element $C \in \mathbb{E}\backslash\{0, 1\}$ for which $g := \overline{S}(C)$ is a generator of $\mathbb{E}^*$, *i.e.*, we have $\mathbb{E}^* = \{g^i \mid 0 \le i < q^n\}$. As $\mathbb{E}$ is a finite field we know that such a generator g exists. Given that $\overline{S}$ is surjective we know that we can find the corresponding $C \in \mathbb{E}\backslash\{0, 1\}$. Now we compute $g_i := \overline{S}(C)^{q^i}$ for $0 \le i < n$. Using any total ordering "$<$", we obtain $g_c := \min\{g_0, \ldots, g_{n-1}\}$ for some $c \in \mathbb{N}$ as the smallest element of this set. One example of such a total ordering would be to use a bijection between the sets $\mathbb{E} \leftrightarrow \{0, \ldots, q^n - 1\}$ and then exploiting the ordering of the natural numbers to derive an ordering on the

elements of the extension field $\mathbb{E}$. Finally, we define $S(X) := [\overline{S}(X)]^{q^c}$ as new affine transformation. To cancel the effect of the Frobenius sustainer, we define $T(X) := [\overline{T}(X)]^{q^{n-c}}$.

Hence, we have now computed a unique normal form for a given private key. Moreover, we can "reverse" these computations and derive an equivalence class of size $n.q^{2n}.(q^n-1)^2$ this way as we have

$$(BX^{q^c} + A, B'X^{q^{n-c}} + A') \bullet (S, \mathcal{P}', T) \text{ for } B, B' \in \mathbb{E}^*, A, A' \in \mathbb{E} \text{ and } 0 \leq c < n\,.$$

□

Remark. To the knowledge of the authors, the additive sustainer for HFE has first been reported in [Tol03]; it was used there for reducing the affine transformations to linear ones. In addition, a weaker version of the above theorem can be found in [WP05b].

For $q = 2$ and $n = 80$, the number of equivalent keys per private key is $\approx 2^{326}$. In comparison, the number of choices for S and T is $\approx 2^{12,056}$. This special choice of parameters has been used in HFE Challenge 1 [Pat96b].

4.1.1 HFE-

We recall that HFE- is the original HFE-class with the minus modification from Section 3.7 applied. In particular, this means that the "shape" of the central polynomial $P'(X')$ is still the same, *i.e.*, all considerations from the previous theorem also apply to HFE-.

Theorem 4.3 *For $K := (S, P, T) \in Aff^{-1}(\mathbb{F}^n) \times \mathbb{E}[X] \times Aff^{-1}(\mathbb{F}^n)$ a private key in HFE and a reduction parameter $r \in \mathbb{Z}^+$ we have*

$$n.q^{2n}(q^n-1)(q^{n-r}-1)\prod_{i=n-r-1}^{n-1}(q^n-q^i)$$

equivalent keys. Hence, the key-space of HFE- can be reduced by this number.

Proof. This proof uses the same ideas as the proof of Theorem 4.2 to obtain a normal form of the affine transformation S, *i.e.*, applying the additive sustainer, the big sustainer and the Frobenius sustainer on this side. Hence, we have a reduction by $n.q^n(q^n-1)$ keys here.

For the affine transformation T, we also have to take the reduction sustainer into account: we use $\tilde{T}(X) : \mathbb{F}^n \to \mathbb{F}^{n-r}$ and fix $\tilde{T}(0) = 0$ by applying the additive sustainer and $\tilde{T}(1) = 1$ by applying the big sustainer, which gives us q^{n-r} and $q^{n-r}-1$ choices, respectively. To avoid double counting with the reduction sustainer, all computations were performed in $\tilde{\mathbb{E}} := \mathrm{GF}(q^{n-r})$ rather than $\mathbb{E}$. Again, we can compute a normal form for a given private key and reverse these computations to obtain the full equivalence class for any given private key in normal form. Moreover, we observe that the resulting transformation $\tilde{T}$ allows for $q^r \prod_{i=n-r-1}^{n-1}(q^n-q^i)$ choices for the original transformation $T : \mathbb{F}^n \to \mathbb{F}^n$ without affecting the output of $\tilde{T}$ and hence, keeping the two fix points $\tilde{T}(0) = 0$ and $\tilde{T}(1) = 1$. Therefore, there are a total of $q^{n-r} \cdot q^r \cdot (q^{n-r}-1) \cdot \prod_{i=n-r-1}^{n-1}(q^n-q^i)$ possibilities for the transformation T without changing the public key equations. Multiplying out the intermediate results for S and T yields the theorem. □

For $q = 2, r = 7$ and $n = 107$, the number of equivalent keys for each private key is $\approx 2^{2129}$. In comparison, the number of choices for S and T is $\approx 2^{23,108}$. This special choice of parameters has been used in Quartz-7m [WP04].

4.1.2 HFEv

Another important variation of Hidden Field Equations is HFEv. In particular, it was used in the signature scheme Quartz [CGP01]. HFEv was introduced in [KPG99]. The HFEv scheme is characterized in the following definition.

Definition 4.4 *Let $\mathbb{E}$ be a finite field with degree n' over $\mathbb{F}$, $v \in \mathbb{Z}^+$ the number of vinegar variables, and P(X) a polynomial over $\mathbb{E}$. Moreover, let $(z_1, \ldots, z_v) := s_{n-v+1}(x_1, \ldots, x_n), \ldots, s_n(x_1, \ldots, x_n)$ for s_i the polynomials of $S(x)$ in multivariate representation and $X' := \phi^{-1}(x'_1, \ldots, x'_{n'})$, using the canonical*

bijection $\phi^{-1} : \mathbb{F}^n \to \mathbb{E}$ *and* $x'_i := s_i(x_1, \ldots, x_n)$ *for* $1 \leq i \leq n'$ *as hidden variables. Then define the central equation as*

$$\begin{aligned} P'_{z'_1,\ldots,z'_v}(X') \quad := \quad & \sum_{\substack{0\leq i,j\leq d \\ q^i+q^j\leq d}} C_{i,j} X'^{q^i+q^j} + \sum_{\substack{0\leq k\leq d \\ q^k\leq d}} B_k(z_1,\ldots,z_v) X'^{q^k} \\ & + A'(z'_1,\ldots,z'_v) \\ & \text{where} \begin{cases} C'_{i,j} X'^{q^i+q^j} & \text{for } C'_{i,j} \in \mathbb{E} \text{ are the quadratic terms,} \\ B'_k(z'_1,\ldots,z'_v) X'^{q^k} & \text{for } B'_k(z'_1,\ldots,z'_v) \text{ depending linearly on } z'_1,\ldots,z'_v \text{ and} \\ A'(z'_1,\ldots,z'_v) & \text{for } A'(z'_1,\ldots,z'_v) \text{ depending quadratically on } z'_1,\ldots,z'_v \end{cases} \end{aligned}$$

and a degree $d \in \mathbb{Z}^+$*, we say the central equations* $\mathcal{P}'$ *are in HFEv-shape.*

The condition that the $B'_k(z'_1, \ldots, z'_v)$ are affine functions (*i.e.*, of degree 1 in the z'_i at most) and $A'(z'_1, \ldots, z'_v)$ is a quadratic function over $\mathbb{F}$ ensures that the public key is still quadratic over $\mathbb{F}$.

Theorem 4.5 *For* $K := (S, P', T) \in \mathit{Aff}^{-1}(\mathbb{F}^n) \times \mathbb{E}[X'] \times \mathit{Aff}^{-1}(\mathbb{F}^m)$ *a private key in HFEv,* $v \in \mathbb{Z}^+$ *the number of vinegar variables,* $\mathbb{E}$ *an* n'*-dimensional extension of* $\mathbb{F}$ *where* $n' := n - v = m$ *we have*

$$n' q^{n+n'+vm} (q^{n'} - 1)^2 \prod_{i=0}^{v-1} (q^v - q^i)$$

equivalent keys. Hence, the key-space of HFEv can be reduced by this number.

Proof. In contrast to HFE-, the difficulty now lies in the computation of a normal form for the affine transformation S rather than the affine transformation T. For the latter, we can still apply the big sustainer and the additive sustainer and obtain a total of $q^m \cdot (q^m - 1) = q^{n'} \cdot (q^{n'} - 1)$ equivalent keys for a given transformation T. Moreover, the HFEv modification does not change the "absorbing behaviour" of the central polynomial P' and hence, the proof from Theorem 4.2 is still applicable.

Instead, we have to concentrate on the affine transformation S here. In order to simplify the following argument, we apply the additive sustainer on S and obtain a linear transformation. This reduces the key-space by q^n. In order to make sure that we do not count the same linear transformation twice, we consider a normal form for the now (linear) transformation S

$$\begin{pmatrix} E_m & F_v^m \\ 0 & I_v \end{pmatrix} \text{ with } E_m \in \mathbb{F}^{m\times m}, F_v^m \in \mathbb{F}^{m\times v}\,.$$

In the above definition, we also have I_v the identity matrix in $\mathbb{F}^{v\times v}$. Moreover, the left-lower corner is the all-zero matrix in $\mathbb{F}^{v\times m}$. The reason for this non-symmetry: we may not introduce vinegar variables in the set of oil variables, but due to the form of the vinegar equations, we can introduce oil variables in the set of vinegar variables. This is done by the following matrix. In particular, for each invertible matrix M_S, we have a unique matrix

$$\begin{pmatrix} I_m & 0 \\ G_m^v & H_v \end{pmatrix} \text{ with an invertible matrix } H_v \in \mathbb{F}^{v\times v}.$$

which transfers M_S to the normal form from above. Again, I_m is an identity matrix in $\mathbb{F}^{m\times m}$. Moreover, we have some matrix $G_m^v \in \mathbb{F}^{v\times m}$. This way, we obtain $q^{vm} \prod_{i=0}^{v-1} (q^v - q^i)$ equivalent keys in the "v" modification alone. As stated previously, the identity matrix I_m ensures that the input of the HFE component is unaltered. However, we do not have such a restriction on the input of the vinegar part and can hence introduce the two matrices G_m^v and H_v: they are "absorbed" into the random terms of the vinegar polynomials $B'_k(z'_1, \ldots, z'_v)$ and $A'(z'_1, \ldots, z'_v)$.

For the HFE component over $\mathbb{E}$, we can now apply the big sustainer to S and obtain a factor of $(q^{n'} - 1)$. In addition, we apply the Frobenius sustainer to the HFE component, which yields an additional factor of

n'. Note that the Frobenius sustainer can be applied both to S and T, and hence, we can make sure that it cancels out and does not affect the degree of the central polynomial $P_{z_1,\ldots,z_v}(X)$. Again, we can reverse all computations and therefore obtain equivalence classes of equal size for each given private key in normal form. □

For the case $q = 2, v = 7$ and $n = 107$, the number of equivalent keys for each private is $\approx 2^{1160}$. In comparison, the number of choices for S and T is $\approx 2^{21,652}$.

4.1.3 HFEv-

Here, we combine both the HFEv and the HFE- modification to obtain HFEv-. In fact, the original Quartz scheme [CGP01] was of this type.

Theorem 4.6 *For $K := (S, P', T) \in Aff^{-1}(\mathbb{F}^n) \times \mathbb{E}[X'] \times Aff^{-1}(\mathbb{F}^{m+v}, \mathbb{F}^{m+r})$ a private key in HFEv-, $v \in \mathbb{Z}^+$ vinegar variables, a reduction parameter $r \in \mathbb{Z}^+$ and $\mathbb{E}$ an n'-dimensional extension of $\mathbb{F}$ where $n' := n - v$ and $n' = m + r$ we have*

$$n' q^{r+2n'+vn'} (q^{n'} - 1)^2 \prod_{i=0}^{v-1} (q^v - q^i) \prod_{i=n'-r-1}^{n'-1} (q^{n'} - q^i)$$

equivalent keys. Hence, the key-space of HFEv- can be reduced by this number.

Proof. This proof is a combination of the two cases HFEv and HFE-. Given that the difficulty for the HFE- modification was in the T-transformation while the difficulty of HFEv was in the S-transformation, we can safely combine the known sustainers without any double-counting. □

For the case $q = 2, r = 3, v = 4$ and $n = 107, n' = 103$, the number of redundant keys is $\approx 2^{1258}$. In comparison, the number of choices for S and T is $\approx 2^{22,261}$. This special choice of parameters has been used in the original version of Quartz [CGP01], as submitted to NESSIE [NES].

4.2 Matsumoto-Imai Scheme A

As HFE, the MIA class uses a finite field $\mathbb{F}$ and an extension field $\mathbb{E}$. However, the choice of the central equation is far more restrictive than in HFE as we only have one monomial here.

Definition 4.7 *Let $\mathbb{E}$ be an extension field of dimension n over the finite field $\mathbb{F}$ with even characteristic and $\lambda \in \mathbb{Z}^+$ an integer with $\gcd(q^n - 1, q^\lambda + 1) = 1$. We then say that the following central equation is of MIA-shape:*

$$P'(X') := X'^{q^\lambda + 1}.$$

The restriction $\gcd(q^n - 1, q^\lambda + 1) = 1$ is necessary first to obtain a permutation polynomial and second to allow efficient inversion of $P'(X')$. In this setting, we cannot apply the additive sustainer as this monomial does not allow any linear or constant terms. Moreover, the monomial requires a factor of one. Hence, we have to preserve this property. As we will see in Section 5, the only sustainers suitable here are the big sustainer, see Section 3.2, and the Frobenius sustainer from Section 3.6.

Remark. In the paper [MI88], MIA was introduced under the name C*. Moreover, it used the branching modifier [WP05c, 4.4] by default. As branching has been attacked very successfully, C* has been used without this modification for any later construction, *e.g.*, [CGP00b, CGP02, CGP00a, CGP03]. However, without the branching condition, the "new" scheme C* coincides with the previously suggested "Scheme A" from [IM85]. To acknowledge this historical development, we decided to come back to the earlier notation and call the scheme presented in this section "MIA" for "Matsumoto-Imai Scheme A". This has been previously suggested in [WP05c].

Theorem 4.8 *For $K := (S, P', T) \in Aff^{-1}(\mathbb{F}^n) \times \mathbb{E}[X'] \times Aff^{-1}(\mathbb{F}^n)$ a private key in MIA we have*

$$n(q^n - 1)$$

equivalent keys. Hence, the key-space of MIA can be reduced by this number.

PROOF. To prove this statement, we consider normal forms of keys in MIA. In particular, we concentrate on a normal form of the affine transformation S where S is in univariate representation. As for HFE and w.l.o.g., let $B := S(1)$ be a non-zero coefficient on Input 1. Unlike HFE we cannot enforce that $S(0) = 0$, so we may have $S(1) = 0$. However, in this case set $B := S(0)$. Applying $\sigma^{-1}(X) := B^{-1}X$ will ensure a normal form for S. In order to "repair" the monomial $P'(X')$, we have to apply an inverse transformation to T. So let $\tau(X) := (B^{q^\lambda+1})^{-1}X$. This way we obtain

$$\begin{aligned}\mathcal{P} &= T \circ \tau^{-1} \circ \tau \circ P' \circ \sigma \circ \sigma^{-1} \circ S \\ &= \tilde{T} \circ (B^{(q^\lambda+1).(-1)}.B^{q^\lambda+1}.X'^{q^\lambda+1}) \circ \tilde{S} \\ &= \tilde{T} \circ P' \circ \tilde{S}\,,\end{aligned}$$

where $\tilde{S}$ is in normal form. In contrast to HFE in Theorem 4.2, we cannot chose the transformations σ and τ independently: each choice of σ implies a particular τ and vice versa. However, the fix point 1 is still preserved by the Frobenius sustainer and so we can apply this sustainer to the transformation S. As for HFE, we compute a normal form for a given generator and a total ordering of $\mathbb{E}$; again, we "repair" the monomial $X'^{q^\lambda+1}$ by applying an inverse Frobenius sustainer to T and hence have

$$(BX^{q^c}, B^{-q^\lambda-1}X^{q^{n-c}}) \bullet (S, P', T) \text{ where } B \in \mathbb{E}^* \text{ and } 0 \leq c < n \text{ for } c \in \mathbb{N}\,,$$

which leads to a total of $n \cdot (q^n - 1)$ equivalent keys for any given private key. Since all these keys form equivalence classes of equal size, we reduced the private key space of MIA by this factor. □

We want to point out that there is also a variation of MIA defined over *odd* characteristic. This variation has been suggested in [WP05c, Sect. 7.1] and uses exactly the same structure for the private key. For technical reasons, the condition on the gcd is replaced by $\gcd(q^n - 1, q^\lambda + 1) = 2$. However, this is irrelevant for our purpose and we have hence the following corollary.

Corollary 4.9 *For $K := (S, P', T) \in Aff^{-1}(\mathbb{F}^n) \times \mathbb{E}[X'] \times Aff^{-1}(\mathbb{F}^n)$ a private key in MIO we have*

$$n(q^n - 1)$$

equivalent keys. Hence, the key-space of MIO can be reduced by this number.

The above corollary can be proved in exactly the same way as Theorem 4.8. In particular, the fact that MIO is defined over odd rather than even characteristic does not impose a restriction in this context.

Remark. Patarin observed that it is possible to derive equivalent keys by changing the monomial P' [Pat96a]. As the aim of this article is the study of equivalent keys by chaining the affine transformations S, T alone, we did not make use of this property. A weaker version of the above theorem can be found in [WP05b]; in particular, it does not take the MIO class into account.

Moreover, we observed in this section that it is not possible for MIA to change the transformations S, T from affine to linear. But Geiselmann *et al.* showed how to reveal the constant parts of these transformations [GSB01]. Hence, having S, T affine instead of linear does not enhance the overall security of MIA.

For $q = 128$ and $n = 67$, we obtain $\approx 2^{469}$ equivalent private keys per class. The number of choices for S, T is $\approx 2^{63,784}$ in this case.

4.2.1 MIA-

We want to point out that MIA itself is insecure, due to a very efficient attack by Patarin [Pat95]. However, for well-chosen parameters q, r, its variation MIA- (also known as C^{*--}) is believed to be secure: as in the case of HFE and HFE-, we use the original MIA scheme and apply the minus modification from Section 3.7.

Theorem 4.10 *For $K := (S, P, T) \in Aff^{-1}(\mathbb{F}^n) \times \mathbb{E}[X] \times Aff^{-1}(\mathbb{F}^n)$ a private key in MIA and a reduction number $r \in \mathbb{Z}^+$ we have*

$$n.(q^n - 1)q^r \prod_{i=n-r-1}^{n-1} (q^n - q^i)$$

equivalent keys. Hence, the key-space of MIA- can be reduced by this number.

PROOF. This proof is similar to the one of MIA, *i.e.*, we apply both the Frobenius and the big sustainer to S and the corresponding inverse sustainer to the transformation T. This way, we "repair" the change on the central monomial $X^{q^\lambda+1}$. All in all, we obtain a factor of $n \cdot (q^n - 1)$ equivalent keys for a given private key.

Next we observe that the reduction sustainer applied to the transformation T alone allows us to change the last r rows of the vector $v_T \in \mathbb{F}^n$ and also the last r rows of the matrix $M_T \in \mathbb{F}^{n\times n}$. This yields an additional factor of $q^r \prod_{i=n-r-1}^{n-1}(q^n - q^i)$ on this side.

Note that the changes on the side of the transformation S and the changes on the side of the transformation T are independent: the first computes a normal form for S while the second computes a normal form on T. Hence, we may multiply both factors to obtain the overall number of independent keys. □

For $q = 128, r = 11$ and $n = 67$, we obtain $\approx 2^{6180}$ equivalent private keys per class. The number of choices for S, T is $\approx 2^{63,784}$ in this case. This particular choice of parameters has been used in Sflash^{v3} [CGP03].

4.3 Unbalanced Oil and Vinegar Schemes

In contrast to the two schemes before, we now consider a class of $\mathcal{MQ}$-schemes which does not mix operations over two different fields $\mathbb{E}$ and $\mathbb{F}$ but only performs computations over the ground field $\mathbb{F}$. Moreover, Unbalanced Oil and Vinegar schemes (UOV) omit the affine transformation T but use $S \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$. To fit in our framework, we set it to be the identity transformation, *i.e.*, we have $T := \tau := id$. UOV were proposed in [KPG99].

DEFINITION 4.11 *Let $\mathbb{F}$ be a finite field and $n, m \in \mathbb{Z}^+$ with $n \geq 2m$. Moreover, let $\alpha'_i, \beta'_{i,j}, \gamma'_{i,j,k} \in \mathbb{F}$. We say that the polynomials below are central equations in UOV-shape:*

$$p'_i(x'_1, \ldots, x'_n) := \sum_{j=1}^{m}\sum_{k=1}^{n} \gamma'_{i,j,k} x'_j x'_k + \sum_{j=1}^{n} \beta'_{i,j} x'_j + \alpha'_i \,.$$

In this context, the variables x'_i for $1 \leq i \leq m$ are called the "vinegar" variables and x'_i for $m < i \leq n$ the "oil" variables. Note that the vinegar variables are combined quadratically while the oil variables are only combined with vinegar variables in a quadratic way. Therefore, assigning random values to the vinegar variables, results in a system of linear equations in the oil variables which can than be solved, *e.g.*, using Gaussian elimination. So the "shape" of UOV is the fact that a system in the oil variables alone is linear. Hence, we may not mix oil variables and vinegar variables in our analysis but may perform affine transformations within one set of these variables. So for UOV, we can apply the additive sustainer and also the Gauss sustainer, introduced in sections 3.1 and 3.5. However, in order to ensure that the shape of the central equations does not change, we have to ensure that the Gauss sustainer influences the vinegar and oil variables separately.

Theorem 4.12 *Let $K := (S, \mathcal{P}', id) \in \mathit{Aff}^{-1}(\mathbb{F}^n) \times \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m) \times \mathit{Aff}^{-1}(\mathbb{F}^m)$ be a private key in UOV. Then we have*

$$q^{n+mn} \prod_{i=0}^{n-m-1} (q^{n-m} - q^i) \prod_{i=0}^{m-1} (q^m - q^i)$$

equivalent keys. Hence, the key-space of UOV can be reduced by this number.

PROOF. As in the case of the schemes before, we compute a normal form for a given private key. First, applying the additive sustainer reduces the affine transformation S to a linear transformation. This results in a factor of q^n in terms of equivalent keys. Second, applying the Gauss sustainer separately within vinegar and oil variables, we can enforce the following structure, denoted $R \in \mathbb{F}^{n\times n}$, on the matrix $M_S \in \mathbb{F}^{n\times n}$ of the (now only) linear transformation S:

$$R := \begin{pmatrix} I_m & 0 & A_m \\ 0 & I_{n-2m} & B_m^{n-2m} \\ 0 & 0 & I_m \end{pmatrix} .$$

In this context, the matrices I_m, I_{n-2m} are the identity elements of $\mathbb{F}^{m\times m}$ and $\mathbb{F}^{(n-2m)\times(n-2m)}$, respectively. Moreover, we have the matrices $A_m \in \mathbb{F}^{m\times m}$ and $B_m^{n-2m} \in \mathbb{F}^{(n-2m)\times m}$. For a given central equation $\mathcal{P}'$, each possible matrix R leads to the same number of equivalent keys. Let

$$E := \begin{pmatrix} F_{n-m} & 0 \\ G_{n-m}^m & H_m \end{pmatrix}$$

be an $(n \times n)$-matrix. Here, we require that the matrices $F_{n-m} \in \mathbb{F}^{(n-m)\times(n-m)}$ and $H_m \in \mathbb{F}^{m\times m}$ are invertible and hence the counting from Lemma 2.4 applies. For $G_{n-m}^m \in \mathbb{F}^{m\times(n-m)}$, we have no restrictions. This way, we define the transformation $\sigma(x) := Ex$ where $x \in \mathbb{F}^n$. Note that these transformations σ form a subgroup within the affine transformations. So we have

$$(Ex + a, id) \bullet (S, \mathcal{P}', id) \text{ for } a \in \mathbb{F}^m \text{ and } E \text{ as defined above.}$$

As this choice of σ partitions the private key space into equivalence classes of equal size, and due to the restrictions on E, we reduced the size of the private key space by an additional factor of $q^{mn}\prod_{i=0}^{n-m-1}(q^{n-m} - q^i)\prod_{i=0}^{m-1}(q^m - q^i)$. □

For $q = 2, m = 64, n = 192$, we obtain $2^{32,956}$ equivalent keys per key — in comparison to $2^{37,054}$ choices for S. If we increase the number of variables to $n = 256$, we obtain $2^{57,596}$ and $2^{65,790}$, respectively. Both choices of parameter have been used in [KPG03].

4.4 Stepwise-Triangular Systems

Unbalanced Oil and Vinegar schemes and Stepwise-Triangular Systems (STS) are quite similar as both are defined over small ground fields rather than ground fields and extension fields. In addition, they enforce a special structure on the input variables. In the case of UOV we have two sets of variables while we use $L \in \mathbb{Z}^+$ such sets in the case of STS, each forming one *layer* or *step*. These layers form a generalized triangular structure, hence the name of these schemes. We capture this intuition more formally below. Stepwise Triangular Schemes were introduced in [WBP04].

DEFINITION 4.13 *Let $n_1, \ldots, n_L \in \mathbb{Z}^+$ be L integers such that $n_1 + \cdots + n_L = n$, the number of variables, and $m_1, \ldots, m_L \in \mathbb{Z}^+$ such that $m_1 + \cdots + m_L = m$, the number of equations. Here n_l represents the number of new variables (step-width) and m_l the number of equations (step-height), both in Step l for $1 \le l \le L$. By convention, we set $n_0 := m_0 := 0$. Now let $\mathcal{P}' \in \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$ be a system of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomials such that the m_l private quadratic polynomials $p'_{m_0+\ldots+m_{l-1}+1}, \ldots, p'_{m_l}$ of each layer l contain only the variables x'_k with $k \le \sum_{j=1}^{l} n_j$, i.e., only the variables defined in all previous steps plus n_l new ones. Then we call $(S, \mathcal{P}', T,) \in Aff^{-1}(\mathbb{F}^n) \times \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m) \times Aff^{-1}(\mathbb{F}^m)$ a private key in* Stepwise Triangular System *shape. If $n_1 = \ldots = n_L = m_1 = \ldots = m_L = r$ for some $r \in \mathbb{Z}^+$, we call this a* regular *Stepwise Triangular System.*

We want to stress in this context that we do not assume any additional structure for the private polynomials $p'_1, \ldots, p'_m$ here. In particular, all coefficients $\gamma'_{i,j,k}, \beta'_{i,j}, \alpha'_i \in \mathbb{F}$ for these polynomials may be chosen at random.

As STS and UOV are based on a similar concept, the following proof on Stepwise Triangular Schemes uses the same ideas as the proof for the UOV class. As for UOV we exploit the fact that we can use Gauss operations within any given layer — and use again the fact that equations of Layer l depend on all variables of the layers $1, \ldots, l$, *i.e.*, we may also perform Gauss operations on these previous layers, as long as the result only affects the given Layer l.

Theorem 4.14 *Let $\mathbb{F}$ be a finite field with $q := |\mathbb{F}|$ elements, $n \in \mathbb{Z}^+$ the number of variables, $m \in \mathbb{Z}^+$ the number of equations and $L \in \mathbb{Z}^+$ the number of layers. Moreover, let $(n_1, \ldots, n_L) \in (\mathbb{Z}^+)^L$ be a vector of integers such that $n_1 + \ldots + n_L = n$ and $m_1, \ldots, m_L \in \mathbb{Z}^+$ integers such that $m_1 + \ldots + m_L = m$. Then for $K := (S, \mathcal{P}', T) \in Aff^{-1}(\mathbb{F}^n) \times \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m) \times Aff^{-1}(\mathbb{F}^m)$ a private key in STS we have*

$$q^{m+n}\prod_{i=1}^{L}\left(q^{n_i(n-\sum_{j=1}^{i} n_j)}\prod_{j=0}^{n_i-1}(q^{n_i} - q^j)\right)\prod_{i=1}^{L}\left(q^{m_i(m-\sum_{j=1}^{i} m_j)}\prod_{j=0}^{m_i-1}(q^{m_i} - q^j)\right)$$

equivalent keys. Hence, the key-space of STS can be reduced by this number.

PROOF. For this proof, we apply both the additive sustainer and the Gauss sustainer. The latter is applied independently on each layer.

First, we observe that we can apply the additive sustainer both to the transformation $S \in \mathrm{Aff}^{-1}(\mathbb{F}^n)$ and $T \in \mathrm{Aff}^{-1}(\mathbb{F}^m)$ to obtain the fix point $S(0) = T(0) = 0$. As a result, we obtain a factor of q^{m+n} and may assume $S \in \mathrm{Hom}^{-1}(\mathbb{F}^n)$ and $T \in \mathrm{Hom}^{-1}(\mathbb{F}^m)$ for the remainder of this proof.

As in the proof of Theorem 4.12, we impose a special structure on the linear transformation S. Therefore, we consider the matrix

$$M_S := \begin{pmatrix} I_{n_1} & * & * & \cdots & & * & * \\ 0 & I_{n_2} & * & & & & * \\ 0 & 0 & I_{n_3} & & & & \\ \vdots & & & \ddots & & & \vdots \\ & & & & I_{n_{L-2}} & * & * \\ 0 & & & & 0 & I_{n_{L-1}} & * \\ 0 & 0 & & \cdots & 0 & 0 & I_{n_L} \end{pmatrix}$$

In $M_S \in \mathbb{F}^{n \times n}$, sub-matrices I_{n_i} are identity matrices in $\mathbb{F}^{n_i \times n_i}$ for $1 \leq i \leq n$. The left lower portion of M_S is zero while the upper right portion of M_S consists of elements of $\mathbb{F}$. To obtain this matrix M_S, we make use of

$$E := \begin{pmatrix} A_{n_1} & 0 & 0 & \cdots & & 0 & 0 \\ * & A_{n_2} & 0 & & & & 0 \\ * & * & A_{n_3} & & & & \\ \vdots & & & \ddots & & & \vdots \\ & & & & A_{n_{L-2}} & 0 & 0 \\ * & & & & * & A_{n_{L-1}} & 0 \\ * & * & & \cdots & * & * & A_{n_L} \end{pmatrix}$$

In this matrix $E \in \mathbb{F}^{n \times n}$, we have invertible components $A_{n_i} \in \mathbb{F}^{n_i \times n_i}$ for $1 \leq i \leq L$. Moreover, the upper right portion of the matrix E is zero while the left lower portion of E consists of elements of $\mathbb{F}$. We see that the above matrix is sufficient to impose this special structure on M_S. Moreover, for each choice of E, we obtain another linear transformation S and hence, M_S is a normal form of S.

Using Lemma 2.4, we can now count the number of possible matrices E and obtain

$$\prod_{i=1}^{L} \left(q^{n_i(n-\sum_{j=1}^{i} n_j)} \prod_{j=0}^{n_i-1} (q^{n_i} - q^j) \right)$$

for the number of possibilities. To see the correctness of the above computation, we specialise it for n_1: here we have the term $\prod_{j=0}^{n_1-1}(q^{n_1}-q^j)$ which computes the number of choices for the matrix A_{n_1} while $q^{n_1(n-n_1)}$ gives the number of choices in the $(n_1 \times (n-n_1))$ column over $\mathbb{F}$ below the matrix A_{n_1}. By induction on n_i we obtain the above formula for $1 \leq i \leq L$. In particular, as M_S is in normal form, there exists exactly one matrix E of the above form for any given $S \in \mathrm{Hom}^{-1}(\mathbb{F}^n)$. Hence, we have established the existence of an equivalence class of this size.

The corresponding proof for the transformation T is analogous, so we can define matrix $E' \in \mathbb{F}^{m \times m}$ similar to matrix E. We only have to replace variables by equations here to reflect the different roles the transformations S and T play. Note that we are allowed to add equations of lower layers to equations of higher layers and hence, may perform the same Gauss operations on equations that we could apply on variables. So we have

$$(Ex + a, E'x + a') \bullet (S, \mathcal{P}', T) \text{ for } a \in \mathbb{F}^n, a' \in \mathbb{F}^m \text{ and } E, E' \text{ defined as above.}$$

As this choice of σ, τ partitions the private key space into equivalence classes of equal size, and due to the restrictions on E, E', we reduced the size of the private key space by the above number. □

Corollary 4.15 *For regular STS with step-width $r \in \mathbb{Z}^+$, $L \in \mathbb{Z}^+$ layers and $n := Lr$ variables, the above formula simplifies to*

$$q^{2n} \left(\prod_{l=1}^{L} q^{r(n-(l-1)r)} \prod_{i=0}^{r-1} (q^r - q^i)^L \right)^2 .$$

Choosing a regular STS scheme and $q = 2, r = 4, L = 25, n = 100$, we obtain $2^{11,315}$ equivalent keys for each given private key. For comparison: the number of choices for the two affine transformations S, T is $2^{20,096}$. Changing the number of layers to 20, and consequently having $r = 5$, we obtain a total of $2^{11,630}$ equivalent keys. These special choices of parameters have been suggested in [KS04].

5 Tightness for MIA and MIO

All theorems in the previous section suffer from the same problem: we do not know if the size-reductions are "tight", *i.e.*, if the sustainers applied are the only ones possible. In this section we proof that for the MIA/MIO class, the big sustainer and the Frobenius sustainer are actually the *only* possible way to achieve equivalent keys for MIA and MIO. We recall that both classes use a finite field $\mathbb{F}$ with $q := |\mathbb{F}|$ elements and an extension field $\mathbb{E}$ of dimension n over $\mathbb{F}$. Over $\mathbb{E}$, they use the monomial $Y' := X'^{q^\lambda+1}$ as central equation for $1 \leq \lambda < n$. While MIA needs q to be even, MIO is defined for q being odd. The proof for the MIA case is based on an unpublished observation by Dobbertin. Its extension to the MIO class is due to the authors.

The starting point of the proof is the following equation which needs to hold for any two equivalent keys for the MIA / MIO class. This is due to the fact that Definition 2.1 restricts us to affine transformations to transfer one private key into. Hence we have the following equation:

$$X^{q^\lambda+1} = T \circ X^{q^\lambda+1} \circ S \,,$$

which we can rewrite as

$$X^{q^\lambda+1} \circ S^{-1} = T \circ X^{q^\lambda+1} \,. \tag{2}$$

We know from Section 2.1 that affine transformations form a group. Moreover, we can use Definition 2.7 to obtain a univariate representation for any given affine transformation. We can hence express (2) as

$$\left(\sum_{i=0}^{n-1} B_i X^{q^i} + A \right)^{q^\lambda+1} = \sum_{i=0}^{n-1} \tilde{B}_i \left(X^{q^\lambda+1} \right)^{q^i} + \tilde{A} \,,$$

for some coefficients $A, \tilde{A}, B_i, \tilde{B}_i \in \mathbb{E}$. Note that we have $(A+B)^p = A^p + B^p$ in a finite field of characteristic p and consequently $(A+B)^q = A^q + B^q$ for $q = p^k$ and some $k \in \mathbb{Z}^+$. We now use a matrix representation of the above equation, similar to the matrix used by Kipnis and Shamir in their cryptanalysis of HFE [KS99]. This yields

$$\begin{pmatrix}
A^{q^\lambda+1} & AB_0^{q^\lambda} X^{q^\lambda} & AB_1^{q^{\lambda+1}} X^{q^{\lambda+1}} & \dots & AB_{n-1}^{q^{\lambda+n-1}} X^{q^{\lambda+n-1}} \\
B_0 A^{q^\lambda} X & B_0^{q^\lambda+1} X^{q^\lambda+1} & B_0 B_1^{q^\lambda} X^{q^{\lambda+1}+1} & & B_0 B_{n-1}^{q^\lambda} X^{q^{\lambda+n-1}+1} \\
B_1 A^{q^\lambda} X^q & B_1 B_0^{q^\lambda} X^{q^\lambda+q} & B_1^{q^\lambda+1} X^{q^{\lambda+1}+q} & \dots & B_1 B_{n-1}^{q^\lambda} X^{q^{\lambda+n-1}+q} \\
\vdots & & \vdots & \ddots & \vdots \\
B_{n-1} A^{q^\lambda} X^{q^{n-1}} & B_{n-1} B_0^{q^\lambda} X^{q^\lambda+q^{n-1}} & B_{n-1} B_1^{q^\lambda} X^{q^{\lambda+1}+q^{n-1}} & \dots & B_{n-1}^{q^\lambda+1} X^{q^{\lambda+n-1}+q^{n-1}}
\end{pmatrix}$$

$$= \begin{pmatrix}
\tilde{A} & 0 & & \dots & 0 \\
0 & \tilde{B}_0^{q^\lambda+1} X^{q^\lambda+1} & 0 & & 0 \\
& 0 & \tilde{B}_1^{q^{\lambda+1}+q} X^{q^{\lambda+1}+q} & & 0 \\
\vdots & & & \ddots & \vdots \\
0 & 0 & 0 & \dots & \tilde{B}_{n-1}^{q^\lambda+(n-1)+q^{n-1}} X^{q^{\lambda+n-1}+q^{n-1}}
\end{pmatrix} (*)$$

As we work in $\mathbb{E}$ which has a multiplicative group of $q^n - 1$ elements, we can reduce all powers larger than or equal to q^n by $q^n - 1$.

Lemma 5.1 *For $\mathbb{F}$ a finite field with $q > 2$ elements, we can only use the big sustainer and the Frobenius sustainer to derive equivalent private keys within the MIA and the MIO class.*

PROOF. For this proof we show that the equations given by $(*)$ imply that $A = 0$ and all B_i for $0 \leq n < n$ except one are zero. Note that $B_0 = \ldots = B_{n-1} = 0$ implies that $S(X)$ is no bijection anymore but the transformation $S(X) = A$ for any input $X \in \mathbb{E}$ and fixed $A \in \mathbb{E}$. Hence, there must exist at least one non-zero coefficient B_i. W.l.o.g., we assume that B_0 is non-zero. Note that this lemma is trivially true for an extension field of degree $n = 1$. Hence, we assume that $\mathbb{E}$ is a proper extension of $\mathbb{F}$ and therefore $n \geq 2$.

For the proof, we make use of the fact that we can reduce all powers in $\mathbb{E}$ by $q^n - 1$. For powers of the form q^i this means that we can reduce the power i by n, *i.e.*, all computations are done in the ring $\mathbb{Z}/n\mathbb{Z}$ and we can hence assume $0 \leq a, b, c, d < n$ in the sequel. Moreover, we can distinguish the following three types of equations in $(*)$:

1. Equations of the form $AB^{q^\lambda + a} + B_b^{q^b} A^{q^\lambda} = 0$ for $a + \lambda \equiv b \pmod{n}$. We call them *equations of type A*. Note that they are related to terms with monomial of the form X^{q^b} for $0 \leq b < n$.

2. Equations of the form $B_a^{q^{\lambda+a}} B_b^{q^b} = 0$ with the condition $a + \lambda \equiv b \pmod{n}$ on the powers. We call them *equations of Hamming weight 1* and say that they are *self-dual*. Note that each row / column in the above matrix contains exactly one equation of Hamming weight 1 and that they correspond to terms with a monomial of the form X^{2q^b} for $0 \leq b > n$. As we have $q > 2$ there is no reduction of the power here.

3. Equations of the form $B_a^{q^{\lambda+a}} B_b^{q^b} + B_c^{q^{\lambda+c}} B_d^{q^d} = 0$ with the following conditions on their powers: first, we have $a \neq b, c \neq d$, as we otherwise would include equations from the diagonal. Obviously, we cannot make the assumption anymore that the right-hand side is equal to zero in this case. Second, we have $a + \lambda \not\equiv b \pmod{n}$ and $c + \lambda \not\equiv d \pmod{n}$ as we obtain equations of Hamming weight 1 otherwise. Third, we need $a + \lambda \equiv d \pmod{n}$ and $c + \lambda \equiv b \pmod{n}$ to ensure that the powers in the monomial $X^{q^b + q^d}$ actually match. We call the pair (a, b) the *dual* of the pair (c, d). Note that this relation is reflexive, *i.e.*, (c, d) is the dual of (a, b). We call these *equations of type B*.

Note that equations of type A and equations of Hamming weight 1 do not mix as we have $q > 2$. Moreover, equations of Hamming weight 1 may not lie on the diagonal as we would have $\lambda + a \equiv a \pmod{n}$ in this case and hence $\lambda \equiv 0 \pmod{n}$, but this violates $0 < \lambda < n$. So far, we did not include any equation from the diagonal in our analysis. We come back to them later.

Inspecting the equation $B_0^{q^\lambda} B_\lambda^{q^\lambda} = 0$ of Hamming weight 1, we see that it implies $B_\lambda = 0$ as we have $B_0 \neq 0$ (see above). In addition, this implies $A = 0$ as we have $AB_0^{q^\lambda} + B_\lambda^{q^\lambda} A^{q^\lambda} = 0$ as an equation of type A. For $n = 2$, we are done. For $n \geq 3$, we can now use all equations of type B of the form $B_0^{q^\lambda} B_b^{q^b} + B_c^{q^{\lambda+c}} B_\lambda^{q^\lambda} = 0$. We notice that we need to meet the following conditions: $b \neq 0, \lambda$ and $c \neq 0, \lambda$ but $c + \lambda \equiv b \pmod{n}$. We see that we can construct pairs (b, c) meeting this conditions for all $b \in \mathbb{Z}/n\mathbb{Z} \backslash \{0, \lambda, 2\lambda\}$ with $0 < b < n$. Using the above equation we have established that all coefficients $B_b = 0$ as $B_0 \neq 0$ and $B_\lambda = 0$. Note that $\lambda \not\equiv 2\lambda \pmod{n}$ as we have $0 < \lambda < n$. Moreover, $2\lambda \not\equiv 0 \pmod{n}$ is not true either, which we see with the following argument: due to the size condition on λ, we know that we need to have $2\lambda = n$ to make the above equation hold. We use the condition $\gcd(q^n - 1q^\lambda + 1) = 1$ for MIA and $\gcd(q^n - 1q^\lambda + 1) = 2$ for MIO to show that $2\lambda = n$ is impossible. Therefore we observe that $(q^{2\lambda} - 1) = (q^\lambda + 1)(q^\lambda - 1)$, *i.e.*, the gcd condition is violated for $n = 2\lambda$.

All left to show is that the coefficient $B_{2\lambda}$ is also equal to zero. To this end, we use the equation $B_{2\lambda}^{q^{3\lambda}} B_0^{q^0} + B_{-\lambda}^{q^0} B_{3\lambda}^{q^{3\lambda}} = 0$ of type B. In order to force the coefficient $B_{2\lambda}$ equal to zero, we need $B_{-\lambda} = 0$ or $B_{3\lambda} = 0$. Therefore, we use the equation $B_{-\lambda} q^0 B_0 q^0 = 0$ of type Hamming weight 1. As we have $B_0 \neq 0$, this implies $B_{-\lambda}$ and hence $B_{2\lambda} = 0$.

We have now established that all coefficients $A = B_1 = \ldots = B_{n-1} = 0$. Using the equations on the diagonal, these conditions also propagate through to the coefficients of the affine transformation T, *i.e.*, to

$\tilde{A}, \tilde{B}_a$ for $0 < a < n$. Given that all coefficients but B_0 are zero, all equations which have terms of the form $B_a B_b$ for $a \neq 0, b \neq 0$ on the left hand side are now also zero, *i.e.*, they do not influence the equations of the form $B_i^{q^{\lambda+i}} B_i^{q^i} = \tilde{B}_j^{q^{\lambda+j}} \tilde{B}_j^{q^j}$ for some i, j with $0 \leq i, j < n$. We can not assume $i = j$ here as the matrix on the right hand side may have been rotated by a constant $r \in \mathbb{Z}^+$ with $0 \leq r < n$. This is equivalent to the application of a Frobenius transformation. Still, we established that S, T may have only one non-zero coefficient in their univariate representation. Therefore, we know that the big sustainer and the Frobenius sustainer are the only two sustainers applicable to $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic systems of the MIA and the MIO type. □

Unfortunately, the above proof is not valid in the case $q = 2$. The reason is that the equations of type A and Hamming weight 1 are mapped to one type of equation, namely $AB_a^{q^{\lambda+a}} + B_b^{q^b} A^{q^\lambda} + B_{a-1}^{q^{\lambda+a-1}} B_{b-1}^{q^{b-1}} = 0$ for $a + \lambda \equiv b \pmod{n}$. All other powers are also reduced $\pmod{n}$. However, as soon as we assume $A = 0$, the above equation collapses to the original equation of Hamming weight 1, and the rest of the proof is again applicable. Alternatively, we could assume that any $B_i = 0$, and derive a similar proof starting with equations of type B. This leads to the following

Corollary 5.2 *For $q = 2$, the affine transformation S in univariate representation either has all coefficients $A, B_0, \ldots, B_{n-1}$ not equal to zero or exactly one coefficient B_i non-equal to zero and all other coefficients equal to zero. The same condition holds for the coefficients $\tilde{A}, \tilde{B}_0, \ldots, \tilde{B}_{n-1}$ of the transformation T.*

Still, we were not able to derive a contradiction with the assumption that all of the above values are non-equal to zero, so we have to leave the proof for the case $q = 2$ as an open problem. However, due to the very high number of equations of $O(n^2)$ compared to only $O(n)$ free variables, we conjecture that the above lemma also holds for $q = 2$ although we expect a far more technical proof in this case.

6 Conclusions

In this article, we showed through the examples of Hidden Field Equations (HFE), Matsumoto-Imai Scheme A (MIA), Unbalanced Oil and Vinegar schemes (UOV), and Stepwise-Triangular Systems (STS) that $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic systems allow many equivalent private keys and hence have a lot of redundancy in their key spaces. These results have been summarized in tables 1 and 2. The first gives an overview on

Table 1: Summary of the reduction results of this article

Scheme (*Section*)	Reduction
UOV (*4.3*)	$q^{n+mn} \prod_{i=0}^{n-m-1} (q^{n-m} - q^i) \prod_{i=0}^{m-1} (q^m - q^i)$
STS (*4.4*)	$q^{m+n} \prod_{i=1}^{L} \left(q^{n_i (n - \sum_{j=1}^{i} n_j)} \prod_{j=0}^{n_i - 1} (q^{n_i} - q^j) \right)$ $\prod_{i=1}^{L} \left(q^{m_i (n - \sum_{j=1}^{i} m_j)} \prod_{j=0}^{m_i - 1} (q^{m_i} - q^j) \right)$
MIA (*4.2*)	$n(q^n - 1)$
MIA- (*4.2.1*)	$n(q^n - 1) q^r \prod_{i=n-r-1}^{n-1} (q^n - q^i)$
HFE (*4.1*)	$nq^{2n}(q^n - 1)^2$
HFE- (*4.1.1*)	$nq^{2n}(q^n - 1)(q^{n-r} - 1) \prod_{i=n-r-1}^{n-1} (q^n - q^i)$
HFEv (*4.1.2*)	$n' q^{n+n'+vm} (q^{n'} - 1)^2 \prod_{i=0}^{v-1} (q^v - q^i)$
HFEv- (*4.1.3*)	$n' q^{r+2n'vn'} (q^{n'} - 1)^2 \prod_{i=0}^{v-1} (q^v - q^i) \prod_{i=n'-r-1}^{n'-1} (q^{n'} - q^i)$

the formulae achieved while the latter features some numerical examples. The symbols used in Table 1 are defined as follows: $n \in \mathbb{Z}^+$ denotes the number of variables, $m \in \mathbb{Z}^+$ is the number of equations, $q := |\mathbb{F}|$ is the number of elements in the ground field $\mathbb{F}$, L the number of layers for STS, and n_l, m_l for $1 \leq l \leq L$ the number of new variables and equations, respectively.

Table 2: Numerical examples for the reduction results of this article

Scheme	Parameters	Choices for S, T (in $\log_2$)	Reduction (in $\log_2$)
UOV	$q = 2, m = 64, n = 192$	37,054	32,956
	$q = 2, m = 64, n = 256$	65,790	57,596
STS	$q = 2, r = 4, L = 25, n = 100$	20,096	11,315
	$q = 2, r = 5, L = 20, n = 100$	20,096	11,630
HFE	$q = 2, n = 80$	12,056	326
HFE-	$q = 2, r = 7, n = 107$	23,108	2129
HFEv	$q = 2, v = 7, n = 107$	21,652	1160
HFEv-	$q = 2, r = 3, v = 4, n = 107$	22,261	1258
MIA	$q = 128, n = 67$	63,784	469
MIA-	$q = 128, r = 11, n = 67$	63,784	6180

We see applications of our results in different contexts. First, they can be used for memory efficient implementations of the above schemes: using the normal forms outlined in this chapter, the memory requirements for the private key can be reduced without jeopardising the security of these schemes. Second, they apply to cryptanalysis as they allow to concentrate on special forms of the private key: an immediate consequence from the existence of the additive sustainers from Section 3.1 is that HFE does not gain any additional strength from the use of affine rather than linear transformations. Hence, this system should be simplified accordingly. Third, constructors of new schemes should keep these sustaining transformations in mind: there is no point in having a large private key space — if it can be reduced immediately by an attacker who can just apply some sustainers. Moreover, the results obtained in this article shine new light on cryptanalytic results, in particular key recovery attacks: as each private key is only a representative of a larger class of equivalent private keys, each key recovery attack can only recover it up to these equivalences as the public key $\mathcal{P}$ cannot contain information about individual private keys but the equivalence class used to construct $\mathcal{P}$.

We want to stress that the sustainers from Section 3 are probably not the only ones possible. We therefore state as an open problem to look for even more powerful transformations. The only case where we know for certain that we found all sustainers possible, is the MIO/MIA class. The corresponding proof can be found in Section 5. We also state as an open problem to find such proofs for the other schemes discussed in this article. In addition, there are other multivariate schemes which could not be discussed in this article, due to space limitations. We are confident that they can be analysed using similar techniques as outlined in this article but have to leave the concrete proof as an open problem.

References

[BCBP03] Alex Biryukov, Christophe De Cannière, An Braeken, and Bart Preneel. A toolbox for cryptanalysis: Linear and affine equivalence algorithms. In *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 33–50. Eli Biham, editor, Springer, 2003.

[BWP05] An Braeken, Christopher Wolf, and Bart Preneel. A study of the security of Unbalanced Oil and Vinegar signature schemes. In *The Cryptographer's Track at RSA Conference 2005*, volume 3376 of *Lecture Notes in Computer Science*. Alfred J. Menezes, editor, Springer, 2005. 13 pages, cf `http://eprint.iacr.org/2004/222/`.

[CGP00a] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Flash: Primitive specification and supporting documentation*, 2000. `https://www.cosic.esat.kuleuven.be/nessie`, submissions, 9 pages.

[CGP00b] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Sflash: Primitive specification and supporting documentation*, 2000. `https://www.cosic.esat.kuleuven.be/nessie`, submissions, Sflash, 10 pages.

[CGP01] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Quartz: Primitive specification (second revised version)*, October 2001. `https://www.cosic.esat.kuleuven.be/nessie` Submissions, Quartz, 18 pages.

[CGP02] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Sflash: Primitive specification (second revised version)*, 2002. `https://www.cosic.esat.kuleuven.be/nessie`, Submissions, Sflash, 11 pages.

[CGP03] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Sflash^{v3}, a fast asymmetric signature scheme — Revised Specification of Sflash, version 3.0*, October 17$^{\text{th}}$ 2003. ePrint Report 2003/211, `http://eprint.iacr.org/`, 14 pages.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *Conference on Applied Cryptography and Network Security — ACNS 2005*, volume 3531 of *Lecture Notes in Computer Science*, pages 164–175. Springer, 2005.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of Hidden Field Equations (HFE) using Gröbner bases. In *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Dan Boneh, editor, Springer, 2003.

[GSB01] W. Geiselmann, R. Steinwandt, and Th. Beth. Attacking the affine parts of SFlash. In *Cryptography and Coding - 8th IMA International Conference*, volume 2260 of *Lecture Notes in Computer Science*, pages 355–359. B. Honary, editor, Springer, 2001. Extended version: `http://eprint.iacr.org/2003/220/`.

[HWyCL05] Yuh-Hua Hu, Lih-Chung Wang, Chun yen Chou, and Feipei Lai. Similar keys of multivariate quadratic public key cryptosystems. In Yvo Desmedt, Huaxiong Wang, Yi Mu, and Yongqing Li, editors, *Cryptology and Network Security, 4th International Conference, CANS 2005, Xiamen, China, December 14-16, 2005, Proceedings*, volume 3810 of *Lecture Notes in Computer Science*, pages 211–222. Springer, 2005.

[IM85] Hideki Imai and Tsutomu Matsumoto. Algebraic methods for constructing asymmetric cryptosystems. In *Algebraic Algorithms and Error-Correcting Codes, 3rd International Conference, AAECC-3, Grenoble, France, July 15-19, 1985, Proceedings*, volume 229 of *Lecture Notes in Computer Science*, pages 108–119. Jacques Calmet, editor, Springer, 1985.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes. In *Advances in Cryptology — EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222. Jacques Stern, editor, Springer, 1999.

[KPG03] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes — extended version, 2003. 17 pages, `citeseer/231623.html`, 2003-06-11.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem. In *Advances in Cryptology — CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 19–30. Michael Wiener, editor, Springer, 1999. `http://www.minrank.org/hfesubreg.ps` or `http://citeseer.nj.nec.com/kipnis99cryptanalysis.html`.

[KS04] Masao Kasahara and Ryuichi Sakai. A construction of public key cryptosystem for realizing ciphtertext of size 100 bit and digital signature scheme. *IEICE Trans. Fundamentals*, E87-A(1):102–109, January 2004. Electronic version: `http://search.ieice.org/2004/files/e000a01.htm\#e87-a,1,102`.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynomial-tuples for efficient signature verification and message-encryption. In *Advances in Cryptology — EUROCRYPT 1988*, volume 330 of *Lecture Notes in Computer Science*, pages 419–545. Christoph G. Günther, editor, Springer, 1988.

[MIHM85] Tsutomu Matsumoto, Hideki Imai, Hiroshi Harashima, and Hiroshi Miyakawa. A cryptographically useful theorem on the connection between uni and multivariate polynomials. *Transactions of the IECE of Japan*, 68(3):139–146, March 1985.

[NES] NESSIE: New European Schemes for Signatures, Integrity, and Encryption. Information Society Technologies programme of the European commission (IST-1999-12324). `http://www.cryptonessie.org/`.

[Pat95] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88. In *Advances in Cryptology — CRYPTO 1995*, volume 963 of *Lecture Notes in Computer Science*, pages 248–261. Don Coppersmith, editor, Springer, 1995.

[Pat96a] Jacques Patarin. Asymmetric cryptography with a hidden monomial. In *Advances in Cryptology — CRYPTO 1996*, volume 1109 of *Lecture Notes in Computer Science*, pages 45–60. Neal Koblitz, editor, Springer, 1996.

[Pat96b] Jacques Patarin. Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology — EUROCRYPT 1996*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Ueli Maurer, editor, Springer, 1996. Extended Version: `http://www.minrank.org/hfe.pdf`.

[PKC05] Serge Vaudenay, editor. *Public Key Cryptography — PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*. Springer, 2005. ISBN 3-540-24454-9.

[Tol03] Ilia Toli. Cryptanalysis of HFE, June 2003. arXiv preprint server, `http://arxiv.org/abs/cs.CR/0305034`, 7 pages.

[WBP04] Christopher Wolf, An Braeken, and Bart Preneel. Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC. In *Conference on Security in Communication Networks — SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 294–309. Springer, September 8–10 2004. Extended version: `http://eprint.iacr.org/2004/237`.

[WHL+05] Lih-Chung Wang, Yuh-Hua Hu, Feipei Lai, Chun-Yen Chou, and Bo-Yin Yang. Tractable rational map signature. In PKC [PKC05], pages 244–257.

[Wol02] Christopher Wolf. *Hidden Field Equations* (HFE) - variations and attacks. Diplomarbeit, Universität Ulm, December 2002. `http://www.christopher-wolf.de/dpl`, 87 pages.

[Wol04] Christopher Wolf. Efficient public key generation for HFE and variations. In *Cryptographic Algorithms and Their Uses 2004*, pages 78–93. Dawson, Klimm, editors, QUT University, 2004.

[Wol05] Christopher Wolf. *Multivariate Quadratic Polynomials in Public Key Cryptography*. Ph.D. thesis, Katholieke Universiteit Leuven, Belgium, November 2005. `http://hdl.handle.net/1979/148`, 156+xxiv pages.

[WP04] Christopher Wolf and Bart Preneel. Asymmetric cryptography: Hidden Field Equations. In *European Congress on Computational Methods in Applied Sciences and Engineering 2004*. P. Neittaanmäki, T. Rossi, S. Korotov, E. Oñate, J. Périaux, and D. Knörzer, editors, Jyväskylä University, 2004. 20 pages, extended version: `http://eprint.iacr.org/2004/072/`.

[WP05a] Christopher Wolf and Bart Preneel. Equivalent keys in HFE, C*, and variations. In *Proceedings of Mycrypt 2005*, volume 3715 of *Lecture Notes in Computer Science*, pages 33–49. Serge Vaudenay, editor, Springer, 2005. Extended version `http://eprint.iacr.org/2004/360/`, 15 pages.

[WP05b] Christopher Wolf and Bart Preneel. Superfluous keys in $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic asymmetric systems. In PKC [PKC05], pages 275–287. Extended version `http://eprint.iacr.org/2004/361/`.

[WP05c] Christopher Wolf and Bart Preneel. Taxonomy of public key schemes based on the problem of multivariate quadratic equations. Cryptology ePrint Archive `http://eprint.iacr.org`, Report 2005/077, 12th of May 2005. `http://eprint.iacr.org/2005/077/`, 64 pages.

[YC04] Bo-Yin Yang and Jiun-Ming Chen. Rank attacks and defence in Tame-like multivariate PKC's. Cryptology ePrint Archive `http://eprint.iacr.org`, Report 2004/061, 29rd September 2004. `http://eprint.iacr.org/`, 21 pages.

The Limit of XL Implemented with Sparse Matrices

Chia-Hsin Owen Chen* Bo-Yin Yang†‡ Jiun-Ming Chen§¶

May 16, 2006

Abstract

System-solving algorithms attracted a lot of attention when Jean-Charles Faugère solved the HFE Challenge 1 using his $\mathbf{F_5}$ algorithm. Today, they are the centerpiece of what is known as *algebraic attacks* and very much a vital technique in cryptology.

Methods that find a Gröbner Bases through the manipulation of extended Macaulay matrices are called Lazard-Faugère solvers. The state-of-the-art $\mathbf{F_5}$ as well as its predecessor $\mathbf{F_4}$, and XL (eXtended Linearization) variants are Gröbner Basis Solvers of the Lazard-Faugère family, which solve a polynomial system by manipulating extended Macaulay matrices. We explore their uses in cracking some multivariate public-key cryptosystems and try to learn how to optimize system-solving in some cryptologically significant situations. In the process, we try to implement and evaluate XL using sparse matrices, and arrive at the conclusion that for generic mid-sized field and systems in the practical range, FXL can potentially outperform more sophisticated $\mathbf{F_4}$-$\mathbf{F_5}$ based methods.

Keywords: system-solving, Gröbner bases, $\mathbf{F_5}$, XL, finite field, optimization.

1 Introduction and Early History

Problem $\mathcal{MQ}$: To solve a polynomial system $l_1(\mathbf{x}) = l_2(\mathbf{x}) = \cdots = l_m(\mathbf{x}) = 0$ of $m \geq n$ (usually quadratic) equations in n variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ over a field $\mathbb{F} = \mathrm{GF}(q)$.

$\mathcal{MQ}$, or Multivariate Quadratic system-solvings is provably NP-hard generically ([18]). Any cryptosystem that deals with many variables in a small finite field and not one big unit depends on its difficulty (including all $\mathcal{MQ}$-schemes a.k.a. multivariates). Among the most prominent of the cryptosystems is the Rijndael Block Cipher, otherwise known as AES.

Algebraic Attacks means loosely the concept of finding algebraic relations in the structure of a cryptosystem, usually solving it in some fashion to break the scheme. While consensus

*Dept. of Electrical Engineering, National Taiwan University, Taipei, Taiwan 106, owenhsin@gmail.com
†Dept. of Mathematics, Tamkang University, Tamsui, Taiwan 251, by@moscito.org
‡TWISC *(Taiwan Information Security Center)* at Nat'l Taiwan U. of Sci. and Tech.
§Dept. of Electrical Eng., Nat'l Cheng-Kung University, Tainan, Taiwan 600, jmchen@cryptomath.com
¶Dept. of Mathematics, National Taiwan University, Taipei, Taiwan

is that Algebraic Attacks do not yet threaten AES [8], it has changed the landscape of cryptology, leading directly to an earthquake in the field of stream ciphers [9].

We discuss modern methods to solve such problems in a cryptology context. Each descends from Buchberger's algorithm of finding lex-ordered Gröbner Bases [5]. This is ubiquitous in symbolic mathematics packages, both commercial (e.g., Maple) and free (e.g., Singular).

Prof. Daniel Lazard [22] suggested tweaking the original Buchberger algorithm by reducing all computations to an elimination on what is called an extended Macaulay matrix [23]. Based on this idea, his student Jean-Charles Faugère proposed a great improvement of Buchberger algorithm called $\mathbf{F_4}$. $\mathbf{F_4}$ has been implemented in MAGMA [24]. A later version, $\mathbf{F_5}$, made headlines 3 years later [17] when it was used to solve the HFE Challenge 1.

Meanwhile, the idea is rediscovered independently in 1999 by Courtois-Klimov-Patarin-Shamir [11] as XL. Courtois *et al* development two useful variants or rather techniques complementary to XL. One, XL2, proved to be a rough equivalent to the idea behind $\mathbf{F_4}$. Fixing (guessing variables) was proved to be a further improvement by Yang *et al*. The other ideas (XFL, XLF and XL') turned out not to have been such great improvements [29, 30].

In the following, we will briefly go over known theory and consensus in the area. Then we will report on some relevant experiments. They have been for the most part in line with theoretical predictions. We also discuss some practical implications on cryptography.

2 The XL, $\mathbf{F_4}$, and $\mathbf{F_5}$ Algorithms

$\mathbf{F_4}$, $\mathbf{F_5}$, and XL algorithms were analyzed in depth recently [2, 3, 11, 12, 15, 16, 17, 28, 29, 30]. In the following, we denote by $\mathbf{x^b}$ the monomial $x_1^{b_1} x_2^{b^2} \cdots x_n^{b_n}$, and its total degree $|\mathbf{b}| = b_1 + \cdots + b_n$. $\mathcal{T} = \mathcal{T}^{(D)} = \{\mathbf{x^b} : |\mathbf{b}| \leq D\}$ is the set of degree-D-or-lower monomials.

2.1 XL

Multiply each equation l_i by all monomials $\mathbf{x^b} \in \mathcal{T}^{(D-2)}$. Reduce as a linear system the equations $\mathcal{R}^{(D)} = \{\mathbf{x^b} l_j(\mathbf{x}) = 0 : 1 \leq j \leq m, |\mathbf{b}| \leq D-2\}$, with the monomials $\mathbf{x^b} \in \mathcal{T}^{(D)}$ as independent variables. Repeat with higher D until we have a solution, a contradiction, or reduce the system to a univariate equation in some variable. The number of monomials will be denoted $T^{(D)} = T = |\mathcal{T}|$, total number of equations $R^{(D)} = R = |\mathcal{R}|$, and the number of independent equations $I^{(D)} = I = \dim(\text{span}\mathcal{R})$.

Proposition 1 ([2, 28]) *The number of monomials is* $T = [t^D] \dfrac{(1-t^q)^n}{(1-t)^{n+1}}$ *which reduces to* $\binom{n+D}{D}$ *when* q *is large. We can then find* $R = R^{(D)} = mT^{(D-2)}$.

Proposition 2 ([28]) *If equations* l_i *are semi-regular, then for all* $D < D_{XL}$, *we have for XL*

$$T - I = [t^D]\, G_{m,n}(t) = [t^D]\, \frac{(1-t^q)^n}{(1-t)^{n+1}} \left(\frac{1-t^k}{1-t^{kq}}\right)^m. \tag{1}$$

The degree of regularity $D_{XL} := \min\{D : [t^D]\, G_{m,n}(t) \leq 0\}$ *is the smallest* D *such that Eq. 1 cannot hold if the system has a solution.* D_0*, the (minimal) operative degree, is never greater than and usually equal to* D_{XL}*. If* $(l_i)_{i=1\cdots m}$ *are* non-semi-regular*,* I *can only decrease.*

Proposition 3 ([12, 29]) $T - I = [t^D]\,((1-t)^{m-n-1}(1+t)^m)$ *for generic quadratic equations if* $D \leq \min(q, D_{XL}^{\infty})$*, where* D_{XL}^{∞} *is the degree of the lowest term with a non-positive coefficient in of* $G_{m,n}^{\infty}(t) = (1-t)^{m-n-1}(1+t)^m$*. For non-generic equations* I *may only decrease.*

Lazard proposed a matrix form [22] but in short we multiply every degree k equation by all monomials up to degree $D-k$, then solve a deg $\leq D$ system linear in all monomials.

2.2 XL2

XL2 is an improved version of XL whose most up-to-date version can be written as follows [29]: Start by tagging each equation with its maximal degree. Run an elimination on the system with monomials in degree-reverse-lex. In the remaining (row echelon form) system, multiply by each variable $x_1, x_2 \cdots$ *all remaining equations with the maximum tagged degree* and eliminate again. When we cannot eliminate all remaining monomials of the maximum degree, increment the operating degree and reallocate more memory.

2.3 $\mathbf{F_4}$ and $\mathbf{F_5}$

We can say that XL/XL2 is a primitive $\mathbf{F_4}$, or (since XL came first) that $\mathbf{F_4}$ and $\mathbf{F_5}$ are an improved version of XL/XL2. In $\mathbf{F_4}$, the matrix is not built at once and there is elimination in between expansion stages, which compresses the number of rows that needs to be handled.

$\mathbf{F_5}$ is a further refinement of $\mathbf{F_4}$. The set of equations is actually build generated one by one (or the matrix row by row). In the process, an algebraic criterion is used to determine, ahead of an elimination process, whether a row will be reduced to zero or not and only the meaningful rows are retained. A complication resulting from the tagging is that the elimination must be done in a strictly ordered way corresponding to no row exchanges ever in a Gaussian.

For complete details of $\mathbf{F_4}/\mathbf{F_5}$, we refer you to [15, 16]. There are two separate concepts of degree here, an apparent operating degree D_{F4} and a higher intrinsic degree. The latter is the degree of the equivalent XL system. XL2 and $\mathbf{F_4}$ have similar operating characteristics and they are shown to be roughly equivalent [29].

Proposition 4 ([3, 29]) *If equations* l_i *are semi-regular,* $\mathbf{F_4}$-$\mathbf{F_5}$ *terminates at operating degree*

$$D_{reg} := \min\left\{D : [t^D]\, \frac{(1-t^q)^n}{(1-t)^n}\, \left(\frac{1-t^k}{1-t^{kq}}\right)^m < 0\right\}.$$

$m-n$	D_{XL}	D_{reg}	$n=9$	$n=10$	$n=11$	$n=12$	$n=13$
0	2^m	m	6.090	46.770	350.530	3322.630	sigmem
1	m	$\lceil\frac{m+1}{2}\rceil$	1.240	8.970	53.730	413.780	2538.870
2	$\lceil\frac{m+1}{2}\rceil$	$\lceil\frac{m+2-\sqrt{m+2}}{2}\rceil$	20.320	2.230	12.450	88.180	436.600

Table 1: System-solving time (seconds): MAGMA 2.12, 2GB RAM, Athlon64x2 2.2GHz

2.4 Current Limitations of Solvers

Table 1 gives some test results with MAGMA 2.12 for generic equations over GF(256):

That we only have 2GB memory is not critical to our inability to solve equations in the realm of 20 byte-sized variable in as many equations, because we also ran into a SIGMEM (out of memory) error with a 15-variable, 15-equation system on a 4-core, 16 GB RAM system.

Table 1 also confirms that guessing is important in generic systems. Simulations by Christopher Wolf [27] establishes at least for the MAGMA implementation of $\mathbf{F_4}$, guessing at the correct number of variables does speed up operations also for GF(2). For larger fields, $\mathbf{F_4}$-$\mathbf{F_5}$ (or XL2) need $m-n \geq 1$, while XL need $m-n \geq 2$. The optimal [29, 30] is often more.

As mentioned before, $\mathbf{F_4}$-$\mathbf{F_5}$ represents the state of the art in system-solving today. In the rest of this text, we will look at ways to speed up a Lazard-Faugère solver.

3 Gaussian and Generalized Gaussian Elimination

According to the description we received from the MAGMA project and Dr. Faugère, even though memory management is very critical, elimination is still relatively straightforward in $\mathbf{F_4}$-$\mathbf{F_5}$, and in the process we see reasonably dense matrices, not extremely sparse ones. There are many ways to solve or reduce a system $A\mathbf{x} = \mathbf{b}$. The best one depends on the context.

3.1 Alternative Elimination Methods

The classical method of equation-solving is suggested by Karl Friedrich Gauss.The time complexity or the number of operations (in particular multiplications) is $\sim N^3/3$ where N is the number of variables or columns. If we are operating with a very unbalanced system of M rows where $M-N$ is large, then [7] the cost is usually given as $\sim M(N^2/2 - M^2/6)$ or $N^2M/3$.

In theory, far better ways to reduce than a straight Gaussian exist, because any method of rapidly multiplying two matrices can be adapted into a method of reducing a matrix, by using Bernstein's Generalized Gaussian Elimination [4]. Berstein showed that if two $N \times N$ matrices can be multiplied in time $\sim \alpha N^\omega$, then an $M \times N$ matrix equation can be reduced in time $\leq \frac{2\alpha(1+\gamma)}{(2^\omega-2)}M^{\omega-1}N + \frac{\alpha M^\omega}{(2^\omega-1)}$, where $\gamma := (7\alpha)/(2^\omega - 4)$. Thus, from the often-cited

Coppersmith-Winograd method which is $\propto N^{2.368}$ or the more common Strassen's Blocking [25] we can build elimination methods. For the latter we can reduce systems in $\sim 4N^{2.8}$ [6].

3.2 Why Simple is Often Best

In these kind of papers, authors often blithely invoke the Coppersmith-Winograd Bound of $N^{2.368}$ without mentioning the context. The CW method is elegant and mathematically beautiful. It is however often neglected in these analyses that those constant factors do count.

To show why this is the case, let's take the Bunch-Hopcroft version of Strassen's Blocking Elimination (which cuts each matrix into equal quarters and do 7 sub-reductions per halving) which is already distinctly non-trivial. The number of multiplications is optimistically $4N^{2.8}$ which means no savings in multiplications until a quarter of a million rows ($N = 2.5 \times 10^5$).

Add bookkeeping overhead and memory locality issues, and we may conclude that Gaussian elimination will dominate more intricate methods until we are almost out of "practical" range even as we approach the hundred-gigabyte machine, especially if we consider this trick (initially due to Jintai Ding to the best of our knowledge and research):

Streaming Gaussian: *Let's take for example* $\mathbb{F} = \mathrm{GF}(2^8)$*, with each element represented by a byte, and each row be stored contiguously in memory. Assume that we wish to conduct the elimination of the first column pivoted at the top left corner. We may facilitate the process by building all 255 multiples of the first row and index them by the leading element. Since we are dealing with a char-2 field, we may now XOR each row with the appropriate multiple of the first row. With appropriate prefetching (today often in hardware) and long SIMD units, a multiplication is now reduced to nearly a single byte written out to memory at streaming speed.*

We give an example of how well this works. A current PC based around a Pentium 4 3GHz CPU can write out to memory at nearly 10GB/s, on a dual channel, double-pumped 400 MHz memory bus, ≈ 0.3 cycles per multiplication. The upcoming Opterons are reputedly capable of doing 16GB/s with the CPU speed at 2GHz, or 1 cycle per 8 multiplications.

A "normal" multiplication (see below) takes a bit more than 3 L1 cache access times (10 cycles for K8, 15 cycles for the P4) for a edge to the Gaussian by a factor of 50–80 or $\approx 2^6$.

This factor gets worse, not better when we work on larger char-2 fields. Suppose we are working with $\mathrm{GF}(2^{16})$ which is stored as two additive bytes. We only need to build a 510-rows-long table (with the leading element being hex 0001, 0002, ... 00FF, then hex 0100, 0200, ..., FF00). Now we can reduce a multiplication to *two* byte of streamed writes, when the multiplications takes about 3.5 times as long (using a form of Karatsuba [20]).

Note that this type of speed-up is not always possible in versions of $\mathbf{F_4}$-$\mathbf{F_5}$ where rows are generated and reduced one by one. So while on GF(2) the elegant $\mathbf{F_5}$ is impressively good, it is hard [on a mid-sized char-2 field] to top the Courtois-Yang variant of XL2 with Streaming Gaussian. An obvious answer is to take advantage of the sparsity of Macaulay matrix.

3.3 Wiedemann for XL

Three well-known methods (all utilizing the existence of Krylov subspaces) adapt to sparse matrices: Conjugate Gradient, Lanczos, and Wiedemann. There are two reasons that we will choose Wiedemann. While Lanczos and CG usually takes about N (as opposed to $3N$) multiplications by the matrix A, they are restricted to symmetric matrices. Another significant issue is that the matrix is not square. Usually R/T is on the order of "a few". If we multiply by A^TA instead, we are giving away half of our advantages already. It is also bothersome that in finite fields there is the problem of self-orthogonal vectors which despite attempted improvements using randomization [14, 21] that we tried to implement, leads to a rate of failure that is way too high – with $n = 10$ or so we can fail more than 1000 times before getting a good answer.

How do we handle this problem? Earlier in 2004, Nicolas Courtois conjectured that we may pick $(1 + \epsilon)T$ of the R equations and have a linearly independent set. Jintai Ding gave heuristic arguments, that if we pick rows at random under the constraint that we have enough equations at each level (this is computable by techniques in [29]), then usually we will have a linearly independent set. We evaluated the pessimistic probability of having a good system as $\gtrsim 1/3$ for T up to hundreds of thousands and $\gtrsim 1/10$ into the billions.

In all our tests in the overwhelming majority of the cases we had a consistent system on the first attempt when the system is solvable and never tested more than twice.

Lastly, Prof. Faugère communicated to us that if we eliminate with a Gaussian the matrices will evolve into non-sparse ones and the same ideas don't apply. So not only are we doing Wiedemann's method for speed, we are also doing it for memory considerations.

4 Implementing Wiedemann for XL

Dr. Douglas Wiedemann [26] proposed this approach to solve a sparse system of equations over finite fields. It is based on the fact that, when a square matrix is repeatedly applied to a vector, the resulting vector sequence is linearly recursive. The problem is to solve the linear system $A\mathbf{x} = \mathbf{b}$. Suppose A is a nonsingular $N \times N$ matrix and $f(z)$ is the minimal polynomial of A. Since A is nonsingular, z is not a factor of $f(z)$, and it can be normalized as $f(z) = c_m z^m + c_{m-1} z^{m-1} + \cdots + c_1 z + 1$. If f is found, we have

$$\mathbf{x} = -(c_m A^{m-1}\mathbf{b} + c_{m-1} A^{m-2}\mathbf{b} + \cdots c_1 \mathbf{b})$$

because $f(A)\mathbf{b} = (c_m A^m + c_{m-1} A^{m-1} + \cdots + c_1 A + I_N)\mathbf{b}$ vanishes.

4.1 A Brief Description of Our Sparse Matrix Solver

Suppose $\mathbf{u}$ is a unit column vector. Since the sequence $\{A^j\mathbf{b}\}_{j=0}$ satisfies the linear recurrence associated to f, the sequence $\left\{\mathbf{u}^T A^j \mathbf{b}\right\}_{j=0}$ also satisfies the linear recurrence $f_{\mathbf{u}}$ which is a proper divisor of f. Therefore, we may invoke the Berlekamp-Massey algorithm to compute $f_{\mathbf{u}}$ from the first $2N$ terms of the sequence $\left\{\mathbf{u}^T A^j \mathbf{b}\right\}_{j=0}$. A schematic of the algorithm is:

1. Compute $A^i\mathbf{b}$ for $j = 0 \cdots 2N - 1$.
2. Set $k = 0$ and $g_0(z) = 1$.
3. Set $\mathbf{u}_{k+1}$ to be the $(k + 1)$-st unit vector
4. Extract the sequence from the result of step 1 (the $(k+1)$-st component of each vector).
5. Apply g_k as a difference operator to this sequence.
6. Run Berlekamp-Massey on the produced sequence and find minimal polynomial $f_{k+1}(z)$.
7. Set $g_{k+1} := f_{k+1}g_k$ and $k := k + 1$. If $\deg(g_k) < N$ and $k < n$, go to step 3.
8. Compute the solution $\mathbf{x}$with $f = g_k$ and the table obtained in step 1,

This takes $2N$ multiplications by A if we have enough space ($2N^2$ slots) to store the vectors, plus the time to run Berlekamp-Massey. 1 multiplication by A takes wN multiplications in $\mathbb{F}$ where w counts the nonzero elements per row ($\lesssim n(n + 3)/2$). When we don't have that space (as is often the case when A is sparse), we may throw away vectors as we build them and recompute, bringing us up to $3N$ multiplications by A.

4.2 Testing Procedure over $\mathrm{GF}(2^8)$

1. Generate a testing quadratic equation set with specific n and m and store it into a file.
2. Read the file and setup system.
3. Raise the system to a certain degree D_{XL} as in Section 2.
4. Randomly select rows to form a new system, $A\mathbf{x} = \mathbf{b}$.
5. Apply Wiedemann's method to solve $\mathbf{x}$.
6. Obtain the solution from the last few elements of $\mathbf{x}$ and check its correctness.

The matrix is in log-form (multiplying in $\mathrm{GF}(2^8)$is typically done with a log table) to save time.

We have also tested systems that are unsolvable (this can be checked via XL with Gaussian) when possible. Note that XL with Wiedemann will only work for $f = m - n \geq 2$ (cf. [29]).

5 Results and Discussion

The theoretical timing for XL with Wiedmann is given by $3rT^2(c_0+c_1 \lg T)$ and $c'T^3/3$. Here r is the bound on the number of non-zero coefficient in each equation, usually $(n+1)(n+2)/2$.

However, as we showed above, c' can be made very small. In practice c_1 is very small and $c_0/c' \sim 2^6$. So in trying to find an optimal solver, we need to take into account these effects.

f	n	3	4	5	6	7	8	9	10	11	12	13
2	D	3	4	4	5	5	6	6	7	7	8	8
	C_{XL}	$2.03 \cdot 10^{-2}$	$1.24 \cdot 10^{-2}$	$2.50 \cdot 10^{-2}$	$1.30 \cdot 10^{-1}$	$4.11 \cdot 10^{-1}$	6.71	$2.87 \cdot 10$	$5.24 \cdot 10^{2}$	$1.58 \cdot 10^{3}$	$2.87 \cdot 10^{4}$	$8.44 \cdot 10^{4}$
	M	$2^{10.0}$	$2^{12.7}$	$2^{14.0}$	$2^{16.5}$	$2^{17.6}$	$2^{20.1}$	$2^{21.0}$	$2^{23.5}$	$2^{24.4}$	$2^{26.8}$	$2^{27.7}$
	M'	$2^{19.9}$	$2^{20.0}$	$2^{20.0}$	$2^{20.2}$	$2^{20.4}$	$2^{21.4}$	$2^{22.1}$	$2^{24.3}$	$2^{25.2}$	$2^{27.5}$	$2^{28.4}$
3	D	3	3	4	4	5	5	6	6	6	7	7
	C_{XL}	$1.56 \cdot 10^{-2}$	$1.56 \cdot 10^{-2}$	$2.05 \cdot 10^{-2}$	$4.53 \cdot 10^{-2}$	$4.08 \cdot 10^{-1}$	1.22	$2.88 \cdot 10$	$8.61 \cdot 10$	$2.89 \cdot 10^{2}$	$4.44 \cdot 10^{3}$	$1.20 \cdot 10^{4}$
	M	$2^{10.2}$	$2^{11.4}$	$2^{14.0}$	$2^{15.1}$	$2^{17.7}$	$2^{18.6}$	$2^{21.2}$	$2^{22.0}$	$2^{22.8}$	$2^{25.3}$	$2^{26.1}$
	M'	$2^{19.9}$	$2^{19.9}$	$2^{20.0}$	$2^{20.0}$	$2^{20.5}$	$2^{20.5}$	$2^{22.2}$	$2^{22.9}$	$2^{23.7}$	$2^{26.1}$	$2^{26.8}$
4	D	3	3	4	4	4	5	5	5	6	6	7
	C_{XL}	$9.40 \cdot 10^{-3}$	$6.20 \cdot 10^{-3}$	$1.88 \cdot 10^{-2}$	$3.76 \cdot 10^{-2}$	$9.84 \cdot 10^{-2}$	1.44	3.39	9.35	$2.87 \cdot 10^{2}$	$5.98 \cdot 10^{2}$	$1.19 \cdot 10^{4}$
	M	$2^{10.5}$	$2^{11.5}$	$2^{14.3}$	$2^{15.3}$	$2^{16.1}$	$2^{18.8}$	$2^{19.6}$	$2^{20.3}$	$2^{22.9}$	$2^{23.7}$	$2^{26.2}$
	M'	$2^{19.9}$	$2^{19.9}$	$2^{20.0}$	$2^{20.1}$	$2^{20.2}$	$2^{20.6}$	$2^{21.1}$	$2^{21.5}$	$2^{23.7}$	$2^{24.5}$	$2^{26.9}$
5	D	3	3	3	4	4	4	5	5	5	6	6
	C_{XL}	$7.80 \cdot 10^{-3}$	$1.57 \cdot 10^{-2}$	$1.41 \cdot 10^{-2}$	$4.23 \cdot 10^{-2}$	$8.72 \cdot 10^{-2}$	$1.91 \cdot 10^{-1}$	3.39	9.69	$2.87 \cdot 10$	$5.98 \cdot 10^{2}$	$1.44 \cdot 10^{3}$
	M	$2^{10.6}$	$2^{11.7}$	$2^{12.6}$	$2^{15.4}$	$2^{16.2}$	$2^{17.0}$	$2^{19.7}$	$2^{20.4}$	$2^{21.1}$	$2^{23.7}$	$2^{24.4}$
	M'	$2^{20.0}$	$2^{20.0}$	$2^{20.0}$	$2^{20.1}$	$2^{20.2}$	$2^{20.3}$	$2^{21.1}$	$2^{21.6}$	$2^{22.1}$	$2^{24.5}$	$2^{25.2}$

Table 2: Final Results for XL-Wiedemann, MS C++ 7; P-D 3.0GHz, 2GB DDR2-533
Pre-allocated memory size: $2^{19.9}$, M: theoretical memory size, M': actual memory size, T: average time in seconds. For small n, the required memory the pre-allocated memory size. As n increases, it requires approximately 1.6 times of the theoretical memory size (due to the memory alignment).

5.1 How Optimized is this Algorithm

We may compute that for $m = 15, n = 13$, $D_{XL} = 8$ and $T = 203490$. We did some 1.3×10^{13} multiplications in GF(2^8). This averages out to a bit less than 20 cycles per multiply.

Since if we arrange the matrix first by rows then by columns we will need to read in a matrix element (log-form), then read a vector component (log-form), then look up the anti-log and add, all the latencies concatenate and we have at least 3 L1 access times plus various arithmetic and logical instructions. For Pentium 4's, the L1 latency is 4 cycles and a tight limit should be about 17 cycles per field multiplication. With some overhead, we are at about 85% of the theoretical optimum. Also note that T before goes up over 10^{24}, the logarithmic term is zero, and it takes a long time to get there.

5.2 Comparison to XL-Gauss

It is long known that [17] for $\mathcal{MQ}$, the equations $\mathcal{R}$ in XL is structured and the top block of the elimination is the rate determining step in XL-Gauss. This let us substitute $T^* = \binom{T+D-1}{D}$ instead of the original T. Mr. Yu-Hua Hu [19] produced these timings: $(m, n) = (10, 8)$: 1 second; $(m, n) = (11, 9)$: 17.9 seconds; $(m, n) = (12, 10)$: 173.11 seconds. He is twice the programmer we are, since his speed is twice that of ours. Indeed, he is running at about 0.38 cycles a multiplication, which is very close to optimal (cf. Section 3.2). There is a high coefficient of linearity dependence so our cycle count estimates are fairly good for larger systems.

However, extrapolating from these numbers, our XL-Wiedemann will match his XL-

Gauss at around $m = 14, n = 12$, and it would take a computer with 26GB of RAM for him to run his program at $m = 15, n = 13$, because an XL-Gauss program would need to hold at least $\binom{20}{8} = 125970$ equations of the top block (actually more) at 200,000 entries a row.

5.3 Possibly Overtaking $\mathbf{F_4}$ (or $\mathbf{F_5}$)

By virtue of having to guess at one fewer variable, $\mathbf{F_4}$-$\mathbf{F_5}$ (or XL2) should enjoy a great advantage when n is small. At the moment, FXL-Wiedemann is able to solve 8 by 8 system (a set of 8 variables in 8 quadratic equations) over GF(2^8)reliably in 4×10^3 seconds, a 9 by 9 in 1.4×10^4 seconds (4 hours) and 10 by 10 in 2.2×10^5 seconds (3 days). This is a lot slower than $F\mathbf{F_4}$, since according to Table 1, a 14 by 14 system can be done in 7.5 days with one guess.

However, we expect that for large enough systems, FXL will eventually be faster even if we discount the memory problem. Asymptotics similar to that of [30] shows that a Lazard-Faugère solver using Gaussian elimination uses time $O(2^{3.4n} \cdot \text{poly}(n))$ on a system with n equations and variables over GF(256) (this is roughly in line with Table 1) and $O(2^{2.8n} \cdot \text{poly}(n))$ when using a Strassen-like solver. Compare this with $O(2^{2.4n} \cdot \text{poly}(n))$ when using a Lanczos-class sparse solver, and we can estimate that FXL will catch up with $F\mathbf{F_4}$ ($\mathbf{F_4}$ with guessing) around $m = n = 19$. Actually, with 3 guesses FXL takes almost exactly 2^{80} cycles for $m = n = 20$. If from the data we have about current $\mathbf{F_4}$-$\mathbf{F_5}$ implementations, we guesstimate that the rate-determining step for $F\mathbf{F_4}$ is a Gaussian-class elimination on the top-degree block at $D = D_{reg}$, then we see a MAGMA-like $\mathbf{F_4}$-based solver with sufficient memory be overtaken around $m = n = 20$, and a $\mathbf{F_4}$-based solver using Strassen-class elimination will be overtaken by FXL (with Wiedemann) around $m = n = 23$.

References

[1] M. Bardet. *Étude des systémes algébriques surdéterminés. Applications aux codes correcteurs et é la cryptographie.* PhD thesis, Universitè Paris VI, Décembre 2004.

[2] M. Bardet, J.-C. Faugère, and B. Salvy. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In *Proc. ICPSS, 2004.*

[3] M. Bardet, J.-C. Faugère, B. Salvy, and B.-Y. Yang, *Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems*, Proc. MEGA'05 (Alghero, 2005).

[4] D. Bernstein, *Matrix Inversion Made Difficult*, preprint at `http://cr.yp.to`.

[5] B. Buchberger *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Innsbruck, 1965.

[6] J. R. Bunch and J. E. Hopcroft, *Triangular Factorizations and Inversion by Fast Matrix Multiplication*, Math. Computations, 24 (1974), p. 231–236.

[7] R. Burden and J. D. Faires, *Numerical Analysis, 7th ed.*, PWS-Kent Publ. Co., 2000.

[8] C. Cid, G. Leurent, *An Analysis of the XSL Algorithm*, Asiacrypt 2005, LNCS 3788, p. 333–345.

[9] N. Courtois and W. Meier, *Algebraic Attacks on Stream Ciphers with Linear Feedback*, Eurocrypt 2003, LNCS 2656, p. 345–359.

[10] D. Coppersmith and S. Winograd, *Matrix Multiplication via Arithmetic Progressions*, J. Symb. Comput. 9 (1990) p. 251–280.

[11] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations*, Eurocrypt 2000, LNCS 1807, p. 392–407.

[12] C. Diem. The XL-Algorithm and a Conjecture from Commutative Algebra. Asiacrypt 2004, LNCS 3329, p. 323–337.

[13] J. Ding, private communication.

[14] W. Eberly, E. Kaltofen, *On Randomized Lanczos Algorithms*, Proc. ISSAC '97, p. 176–183, ACM Press '97.

[15] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999. Proc. MEGA'98 (Saint-Malo, 1998).

[16] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero F5. In T. Mora, editor, *Proceedings of ISSAC*, p. 75–83. ACM Press, July 2002.

[17] J.-C. Faugère and A. Joux. Algebraic cryptanalysis of Hidden Field Equation (HFE) cryptosystems using Gröbner bases. Crypto 2003, LNCS 2729, p. 44–60.

[18] M. Garey and D. Johnson, *Computers and Intractability, A Guide to the Theory of NP-completeness*, W. H. Freeman, NYC 1979.

[19] Y.-H. Hu, private communication.

[20] A. Karatsuba and Yu. Ofman, *Multiplication of Many-Digital Numbers by Automatic Computers*, Doklady Akad. Nauk SSSR 145(1962), p. 293-294. Translation in Physics-Doklady 7(1963), p. 595-596.

[21] B. LaMacchia and A. Odlyzko, *Solving Large Sparse Linear Systems over Finite Fields*, Crypto '90, LNCS 537, p. 109–133.

[22] D. Lazard. Gaussian Elimination and Resolution of Systems of Algebraic Equations. EUROCAL 83, LNCS 162, p. 146–157.

[23] F. S. Macaulay. *The algebraic theory of modular systems.*, vol. xxxi, *Cambridge Mathematical Library.* Camb. Univ. Press, 1916.

[24] MAGMA Computational Algebra System (`http://magma.maths.usyd.edu.au/`).

[25] V. Strassen, *Gaussian Elimination is not Optimal*, Numer. Math. 13 (1969) p. 354-356.

[26] D. Wiedemann, *Solving Sparse Linear Equations over Finite Fields*, IEEE Trans. on Info. Theo., v. IT-32 (1976), no. 1, p. 54–62.

[27] C. Wolf, private communication.

[28] B.-Y. Yang and J.-M. Chen, *Theoretical Analysis of XL over Small Fields*, ACISP 2004, LNCS 3108, p. 277-288.

[29] B.-Y. Yang and J.-M. Chen, *All in the XL Family: Theory and Practice*, ICISC 2004, LNCS 3506, p. 67–86.

[30] B.-Y. Yang, J.-M. Chen and N. Courtois, *On Asymptotic Security Estimates in XL and Gröbner Bases-Related Algebraic Cryptanalysis*, ICICS 2004, LNCS 3269, p. 401-413.

Zhuang-Zi: A New Algorithm for Solving Multivariate Polynomial Equations over a Finite Field

Jintai Ding[1], Jason E. Gower[1] and Dieter S. Schmidt[2]

Department of Mathematical Sciences[1]
Department of Electrical & Computer Engineering and Computer Science[2]
University of Cincinnati
Cincinnati, OH 45220
USA
ding@math.uc.edu, gowerj@math.uc.edu, dieter.schmidt@uc.edu

Abstract. We present the Zhuang-Zi algorithm, a new method for solving multivariate polynomial equations over a finite field. We describe the algorithm and present examples, some of which cannot be solved with the fastest known algorithms.

Keywords: multivariate polynomials, Hidden Field Equation, polynomial roots

1 Introduction

Solving a single variable polynomial equation or a set of multivariate polynomial equations has always been at the center of the development of mathematics. It is not only inspired by our curiosity, but also by the ubiquitous role these simple but fundamental problems play in all branches of science.

Though the Babylonians did not invent the notion of an "equation," they found the first algebraic solution to problems, which gave rise to what we call a single variable quadratic equation today. The first known solution of this problem is given in the Berlin papyrus from the Middle Kingdom of Egypt (circa 2160–1700 BC) [Smi52].

Though the first great success is surely solving the single variable quadratic equation, the next successes came much later with Ferro solving the single variable cubic equation and Ferrari solving the single variable quartic in the 16th century. However Galois' theory put an end to the hope of finding an elegant algebraic formula for higher order single variable equations.

The situation is very different in the multivariate case. The real great success is the Gröbner basis method [Buc65], which comes from the ideas of modern algebraic geometry. Solving polynomial equations over the integers is also a very interesting direction, for example Fermat's last theorem, but it is a completely different story which we will omit here.

A very different and modern direction is to solve multivariate equations over a finite field. Recently much effort in this area has been inspired by the appearance of multivariate public key cryptography. For the single variable case, there are very efficient algorithms, such as the Berlekamp algorithm, for factoring polynomials of relatively low degree. For the multivariate case, one can also use an extension of the Gröbner basis.

The idea of a public key cryptosystem was first suggested by Diffie and Hellman. The first practical implementations were the Diffie–Hellman Protocol, which uses discrete logarithm, and RSA, which uses the product of two large prime numbers. Public key cryptography allows any two parties to communicate securely over an open communication channel, like the Internet, and it now plays a fundamental role in our communication systems. However recent developments in quantum computing, in particular Peter Shor's polynomial-time integer factorization algorithm, shows that a quantum computer can be used to break RSA. Thus there has been great interest in constructing other public key cryptosystems.

One alternative is to use multivariate polynomials, and in particular quadratic polynomials. The security of such constructions is suggested by the proven theorem that solving a set of multivariate polynomial equations over a finite field is, in general, an NP-hard problem [GJ79]. Nevertheless, this result is not enough to guarantee the security of such a cryptosystem.

Recent research in multivariate public key cryptography has stimulated a search for new methods for solving multivariate polynomial equations over a finite field, along the line of Gröbner bases. Examples include XL [CKPS00] and the enhanced Gröbner bases methods F_4 and F_5 of Faugère [Fau99,Fau02]. Inspired by the work in this area, we propose a new algorithm to solve a set of multivariate polynomial equations over a finite field.

2 Background

Let k be a finite field with q elements and suppose we have m polynomials $f_0, f_1, \ldots, f_{m-1} \in k[x_0, x_1, \ldots, x_{n-1}]$. We wish to find all $(a_0, a_1, \ldots, a_{n-1}) \in k^n$, such that

$$\begin{aligned} f_0(a_0, a_1, \ldots, a_{n-1}) &= 0 \\ f_1(a_0, a_1, \ldots, a_{n-1}) &= 0 \\ &\vdots \\ f_{m-1}(a_0, a_1, \ldots, a_{n-1}) &= 0 \end{aligned} \tag{1}$$

We may as well work in the ring

$$k[x_0, x_1, \ldots, x_{n-1}]/(x_0^q - x_0, x_1^q - x_1, \ldots, x_{n-1}^q - x_{n-1}),$$

though for convenience we will abuse notation and write $k[x_0, x_1, \ldots, x_{n-1}]$. The key idea of our new algorithm is to shift perspectives from the space of polynomials $k[x_0, x_1, \ldots, x_{n-1}]$ with coefficients in the small field k, to a space of polynomials $K[X]$ with coefficients in some suitably chosen extension field K.

To simplify matters, let us assume that $m = n$. Choose any irreducible polynomial $g(y) \in k[y]$ of degree n. Then $K = k[y]/(g(y))$ is a degree n field extension of k. Let ϕ be the standard k-linear map that identifies K with the n-dimensional vector space k^n, i.e., $\phi : k^n \longrightarrow K$, defined by

$$\phi(a_0, a_1, \ldots, a_{n-1}) = a_0 + a_1 y + \cdots + a_{n-1} y^{n-1} \tag{2}$$

Let $f : k^n \longrightarrow k^n$ be the polynomial map defined by $f = (f_0, f_1, \ldots, f_{n-1})$. We can lift f up to the extension field K using ϕ to create a map $F : K \longrightarrow K$ defined by

$$F = \phi \circ f \circ \phi^{-1}.$$

Using the Lagrangian interpolation formula, we can think of F as a polynomial in $K[X]$, where X is an intermediate. In fact, F has a unique representation in the quotient space $K[X]/(X^{q^n} - X)$. For any given f, the corresponding F can be calculated by solving a set of linear equations. The following theorem tells us the exact form of this representation.

Theorem 1. *Using the notation as defined above, for a linear polynomial map $f = (f_0, f_1, \ldots, f_{n-1})$ we have*

$$F(X) = \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \mod (X^{q^n} - X),$$

for some $\beta_i, \alpha \in K$. If f is a quadratic polynomial map, then

$$F(X) = \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} \gamma_{ij} X^{q^i + q^j} + \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \mod (X^{q^n} - X),$$

for some $\gamma_{ij}, \beta_i, \alpha \in K$. Representations for higher order polynomial maps are similarly described. In the case of $q = 2$, the formulas are slightly different.

In this paper, we will identify the map F with its corresponding representation given in Theorem 1.

It is now clear that we can move freely between multivariate functions and single variable functions, and we will do so in order to solve the original system of equations. This is the basic idea of Matsumoto-Imai, Patarin, Kipnis and Shamir [MI88,Pat00,KS99], and is also the basis of our algorithm. Given a system of equations such as (1), the basic strategy will be to lift the associated polynomial map f to the map F in the extension field K. The roots of the representation of F given in Theorem 1 correspond exactly with the solutions to the original system of equations defined over k. Once we have the roots in K, we can descend down to k^n with ϕ^{-1}. It remains to develop techniques for reducing the degree of F, which if successful, will allow us to use efficient algorithms for solving single variable polynomial equations.

We note a fundamental difference between our algorithm and others which is that ours can be used only with finite fields and cannot be used with fields of

characteristic zero. The reason for this is that the lifting from the multivariate system to a single variable equation works very well for a finite field, but not for characteristic zero cases. However, in the case of finite fields, our algorithm unifies the two problems of solving single variable and multivariate polynomial equations into a single problem. We have named this algorithm after Zhuang-Zi, an ancient Chinese philosopher who we believe was one of the first to propose the idea of shifting from a local view of problems to a global view.

The remainder of this paper is organized as follows. We will explain the basic algorithm, including a method to reduce the degree of F, and present a toy example in order to show how the algorithm works. We then present more meaningful examples and conclude with a discussion of future work.

3 The Zhuang-Zi Algorithm

We will start with the standard case of $m = n$, where we have the same number of variables and equations. The Zhuang-Zi algorithm takes the polynomials $f_0, f_1, \ldots, f_{n-1} \in k[x_0, x_1, \ldots, x_{n-1}]$ and a positive integer D as its input, where D is the upper bound on the degree of a polynomial equation which can be solved efficiently. When successful the algorithm returns all n-tuples $(a_0, a_1, \ldots, a_{n-1}) \in k^n$ such that $f_i(a_0, a_1, \ldots, a_{n-1}) = 0$, for $i = 0, 1, \ldots, n-1$.

Step 1: Choose any degree n irreducible polynomial $g(y) \in k[y]$ and define $K = k[y]/(g(y))$. Let $\phi : k^n \longrightarrow K$ be as defined in (2). Define $f = (f_0, f_1, \ldots, f_{n-1})$, lift this to K by $F = \phi \circ f \circ \phi^{-1}$, and compute the polynomial representation of $F(X)$ modulo $X^{q^n} - X$. If the $\deg(F(X)) \leq D$, then go to the last step; otherwise continue to the next step.

Step 2: Let $G = \mathrm{Gal}(K/k)$ be the Galois group of K over k consisting of the Frobenius maps $G_i(X) = X^{q^i}$, for $i = 0, 1, \ldots, n-1$. Calculate

$$F_i(X) = G_i \circ F(X) = F(X)^{q^i} \mod (X^{q^n} - X),$$

for $i = 0, 1, \ldots, n-1$. Note that $F_0(X) = F(X)$.

Step 3: Let N be the total number of monomials that appear in any $F_i(X)$. For each $F_i(X)$ create a row vector in K^N, where the entries are the coefficients of $F_i(X)$ listed in decreasing order, and construct an $n \times N$ matrix using these row vectors. Then use Gaussian elimination to produce a new set of t basis polynomials $S = \{S_0(X), S_1(X), \ldots, S_{t-1}(X)\}$. In other words eliminate the monomials in the order of the highest degree first. Label the elements of S so that $S_{t-1}(X)$ is the element of lowest degree. If $\deg(S_{t-1}(X)) \leq D$, then go to the last step; otherwise continue to the next step.

Step 4: It must be that the polynomial of minimal degree in S has degree greater than D. For each $i = 0, 1, \ldots, t-1$ and $j = 0, 1, \ldots, n-1$ compute

$$X^{q^j} S_i(X) \mod (X^{q^n} - X).$$

As before, apply Gaussian elimination to the matrix associated with this set of polynomials to produce a set S' of new basis polynomials. Let $S'_{t'-1}(X)$ be the polynomial in S' of minimal degree. If $\deg(S'_{t'-1}(X)) \leq D$, then go to the last step; otherwise replace S with S' and repeat this step.

Step 5: At this point we have a polynomial $G(X)$ with $\deg(G(X)) \leq D$. Find the roots of $G(X) = 0$ with a suitable method to obtain a set $W = \{\alpha \in K \mid G(\alpha) = 0\}$. The solutions of $F(X) = 0$ will be the subset $\{\alpha \in W \mid F(\alpha) = 0\}$.

Remark 1. Since the complexity of any polynomial root finding method depends on the degree of the given polynomial and the size of the field, so too does the complexity of the Zhuang-Zi algorithm. Improvements in the area of polynomial root finding methods will translate directly into an improvement for the Zhuang-Zi algorithm. It is possible to run several such algorithms in parallel on intermediate polynomials, while at the same time the Zhuang-Zi algorithm continues to find polynomials of lower degrees. When one process finds the roots then the entire computation can be stopped.

Remark 2. A straight forward method for finding the roots of $F(X) = 0$ is to make $F(X)$ square free and then to compute

$$\gcd(X^{q^n} - X, F(X)),$$

which will return the product of all linear factors [Knu81]. The answer can then be obtained quickly from it, in particular when $F(X)$ has only one or a few linear factors. The only difficulty is that the direct approach is time consuming since q^n can be very large, and it is better to use the Frobenius operation $X^q \bmod F(X)$ and repeated squaring before finding the gcd. More efficient methods for finding the roots of a polynomial in a finite field exist and they are described for example in [GCL92,LN03,vG03].

Remark 3. If f consists of quadratic polynomials then the only powers of X that may arise in Steps 1–3 are either of the form X^{q^i} or $X^{q^i+q^j}$. Each application of Step 4 then multiplies these monomials by X^{q^l}. At the first application of Step 4 terms of of the form $X^{q^i+q^j+q^l}$ appear, that is third order terms in $k[x_0, \ldots, x_n]$, and at the next iteration fourth order terms and so on. All possible monomials will have been generated after no more than nq steps, since there are only nq possible monomials in $K[X]/(X^{q^n} - X)$.

Remark 4. If no k-linear combination of the polynomials $f_0, f_1, \ldots, f_{n-1}$ is zero in $k[x_0, x_1, \ldots, x_{n-1}]$ (modulo $x_0^q - x_0, x_1^q - x_1, \ldots, x_{n-1}^q - x_{n-1}$), then no K-linear combination of the polynomials $F_0, F_1, \ldots, F_{n-1}$ is zero in $K[X]$ (modulo $X^{q^n} - X$), since such a linear combination of the F_i will have degree at most $q^n - 1$.

Remark 5. The Zhuang-Zi algorithm works also when $m \neq n$. In this case one has to use the maximum of m and n. When $m < n$, there are fewer equations than variables and one simply introduces $n - m$ polynomials identical to 0. If there are more equations than variables ($m > n$) then one can simply introduce $m - n$ fictitious variables $x_n, \ldots, x_{m-1}$.

Remark 6. It should be noted that if m is too small then we know that there will be a large number of solutions (roughly q^{n-m-1}), and therefore the polynomials in the ideal generated by the $F_i(X)$ will have high degree. If the Zhuang-Zi algorithm is unable to generate a polynomial of sufficiently small degree, then we may never reach the factorization step.

4 Examples

We present an illustrative example to see how the Zhuang-Zi algorithm works in practice. We then present two non-trivial examples where Zhuang-Zi succeeds and Gröbner bases fail.

4.1 A Toy Example

In order to show how the algorithm works we first present a simple example of two quadratic equations in two variables with coefficients in the field $k = GF(2^2)$. We define the polynomial map $f : k^2 \longrightarrow k^2$ by its components

$$f_0(x_0, x_1) = x_0^2 + x_1 + 1$$
$$f_1(x_0, x_1) = x_1^2 + x_0 x_1 + 1$$

in $k[x_0, x_1]$.

The nonzero elements in the field $k = GF(4)$ form a multiplicative group, which is generated by an element that we denote by a. The addition and multiplication table for elements in $GF(4)$ can be written in terms of a as follows.

$+$	0	1	a	a^2
0	0	1	a	a^2
1	1	0	a^2	a
a	a	a^2	0	1
a^2	a^2	a	1	0

$*$	0	1	a	a^2
0	0	0	0	0
1	0	1	a	a^2
a	0	a	a^2	1
a^2	0	a^2	1	a

One irreducible polynomial of degree two with coefficients in k is

$$g(y) = y^2 + y + a^2.$$

The mapping $\phi : k^2 \longrightarrow K$ is defined by

$$\phi(x_0, x_1) = x_0 + x_1 y = X,$$

while $\phi^{-1} : K \longrightarrow k^2$ is defined by (using matrix notation)

$$\phi^{-1}(X) = (X, X^4) \begin{pmatrix} 1+y & 1 \\ y & 1 \end{pmatrix} = (x_0, x_1)$$

With this notation, the polynomial map $F = \phi \circ f \circ \phi^{-1}$ is given by

$$F(X) = yX^8 + yX^5 + X^4 + X^2 + X + y + 1.$$

Since this is a trivial example, we could factor $F(X)$ directly and obtain

$$F(X) = y(X+y)(X^3+X^2+1)(X^4+(y+1)X^3+aX^2+(ay+1)X+a^2)$$

from which we can see that $X = y$ is the only solution of the equation $F(X) = 0$ in K. If we write $y = 0+1\cdot y$, we see that this solution corresponds to the solution $(x_0, x_1) = (0, 1) \in k^2$ of the system $f_0 = f_1 = 0$. In general we would expect the degree of $F(X)$ to be much larger, and so we have designed our method to reduce the degree to a reasonable level that can be handled by an efficient polynomial root finding method such as the distinct degree factorization.

In order to illustrate how the algorithm works, we compute the functions $F_i(X) = F(X)^{q^i} \bmod (X^(16) - X)$ for $i = 0, 1$, noting that here $n = 2$ and $q = 4$. So we have

$$\begin{aligned} F_0(X) &= F(X) = yX^8 + yX^5 + X^4 + X^2 + X + y + 1 \\ F_1(X) &= F^4(X) = X^8 + (y+1)X^5 + X^4 + (y+1)X^2 + X + y \end{aligned}$$

We then create a 2×9 matrix with ij^{th} entry equal to the coefficient of X^{8-j} in $F_i(X)$, where $i = 0, 1$ and $j = 0, 1, \ldots, 8$. This matrix is brought into row echelon form via the Gaussian algorithm, which produces the new polynomials $S_0(X)$ and $S_1(X)$.

$$\begin{aligned} S_0(X) &= X^8 + (a^2y+1)X^4 + a^2yX^2 + (a^2y+1)X + a^2 + 1 \\ S_1(X) &= X^5 + (y+a^2)X^4 + (y+a)X^2 + (y+a^2)X + y + a \end{aligned}$$

In order to reduce the degree further, we now multiply $S_0(X)$ and $S_1(X)$ each by X and X^4, and then add these additional four polynomials to the given set of polynomials. Once again Gaussian elimination is applied and produces the set of six polynomials $S_0(X), S_1(X), \ldots, S_5(X)$.

$$\begin{aligned} S_0 &= X^{12} + (ay+1)X^3 + (ay+a)X^2 + a^2yX + a \\ S_1 &= X^9 + (y+1)X^3 + (a^2y+a^2)X^2 + (a^2y+a^2)X + ay + a^2 \\ S_2 &= X^8 + aX^3 + (a^2y+a^2)X^2 + (y+a)X + a^2y \\ S_3 &= X^6 + aX^3 + (a^2y+1)X^2 + a^2yX + a^2y + a \\ S_4 &= X^5 + (ay+a)X^3 + (ay+1)X^2 + (ay+1)X + a^2 \\ S_5 &= X^4 + (ay+1)X^3 + (a^2y+a)X^2 + a^2X + y + a^2 \end{aligned}$$

Since the degree reduction was not significant, we repeat the process. Multiplying each $S_i(X)$ by X and X^4, adding these polynomials to the associated matrix, and then applying Gaussian elimination reduces the set of eighteen poly-

nomials to eleven polynomials $S_0(X), S_1(X), \ldots, S_{10}(X)$.

$$\begin{aligned}
S_0(X) &= X^{13} + (y+a)X + a \\
S_1(X) &= X^{12} + (y+1)X + ay + a \\
S_2(X) &= X^{10} + a^2yX + a^2y \\
S_3(X) &= X^9 + (y+1)X + a^2y \\
S_4(X) &= X^8 + (a^2y+1)X + a^2y \\
S_5(X) &= X^7 + (ay+a)X + a^2y + a^2 \\
S_6(X) &= X^6 + (a^2y+1)X + y + a \\
S_7(X) &= X^5 + yX + y \\
S_8(X) &= X^4 + (ay+1)X + ay \\
S_9(X) &= X^3 + (a^2y+1)X + 1 \\
S_{10}(X) &= X^2 + (y+1)X + y
\end{aligned}$$

The factorization of $S_{10}(X) = (X+1)(X+y)$ shows that spurious solutions can appear, and must be screened for and discarded. If the process were to be repeated, the set of fourteen polynomials $S_0(X), S_1(X), \ldots, S_{13}(X)$ will be generated and the spurious solution will disappear.

$$\begin{aligned}
S_0 &= X^{14} + ay + a, \\
S_1 &= X^{13} + a^2y + 1, \\
S_2 &= X^{12} + ay + 1, \\
S_3 &= X^{11} + ay, \\
S_4 &= X^{10} + a, \\
S_5 &= X^9 + a^2y + a^2, \\
S_6 &= X^8 + y + a, \\
S_7 &= X^7 + a^2y + a, \\
S_8 &= X^6 + a^2y, \\
S_9 &= X^5 + a^2, \\
S_{10} &= X^4 + y + 1, \\
S_{11} &= X^3 + ay + a^2, \\
S_{12} &= X^2 + y + a^2, \\
S_{13} &= X + y.
\end{aligned}$$

As expected each polynomial has the factor $X + y$. The set of polynomials as given is now invariant when our process is again applied, which must eventually occur since we are working in a finite field.

Of course this simple example could have been solved more easily by finding the Gröbner basis $\{x_0, x_1 + 1\}$ for the equations

$$\begin{aligned} f_0(x_0, x_1) &= 0 \\ f_1(x_0, x_1) &= 0 \\ x_0^4 - x_0 &= 0 \\ x_1^4 - x_1 &= 0 \end{aligned}$$

4.2 Generating Non-Trivial Examples

The Zhuang-Zi algorithm works of course for linear systems of equations and can give insight into how subspaces of k^n are represented in the bigger field K. Nevertheless, the main application is to the nonlinear multivariate problem where Gröbner bases methods do not succeed. As mentioned earlier, the Zhuang-Zi algorithm requires that we work in a finite field, whereas Göbner bases do not. When a Gröbner basis is computed in a finite field, it is accomplished usually by augmenting the original set of equations with those defining the finite field.

Examples that can be solved easily by the Zhuang-Zi algorithm, but only with great difficulties via Gröbner bases, can be constructed easily. The idea is to select a function $F(X) : K \longrightarrow K$ of low enough degree, so that it can be factored easily, while the corresponding mapping $f : k^n \longrightarrow k^n$ must be complicated. The degree of the components $f_0, f_1, \ldots, f_{n-1}$ of f depends on which powers of X have been selected in F. Terms in F of the form X^{q^i} give rise to linear terms in the components of f, $X^{q^i+q^j}$ leads to quadratic terms, $X^{q^i+q^j+q^l}$ leads to cubic terms, and so on. By keeping the exponents $i, j, l, \ldots$ in $X^{q^i}, X^{q^i+q^j}, X^{q^i+q^j+q^l}, \ldots$ small, we can choose a polynomial F of low enough degree so as to be easily factored, while the corresponding components of f will be quadratic, cubic, or higher degree polynomials in $k[x_0, x_1, \ldots, x_{n-1}]$. This idea is reminiscent of what has been suggested for the HFE public key encryption scheme [Pat96]. We now generate such an example.

Let $k = GF(2^3)$ and let $K = k[y]/(g(y))$ be a degree n extension of k, for some irreducible $g(y) \in k[y]$. We use a polynomial of low degree in $K[X]$:

$$F(X) = X^{72} + a_1 X^{65} + a_2 X^{64} + a_3 X^{16} + a_4 X^9 + a_5 X^8 + a_6 X^2 + a_7 X + a_8, \quad (3)$$

where the coefficients a_j, for $j = 1, \ldots, 8$, are chosen at random from k, treated as a subfield of K via the standard embedding. With $q = 8$, all powers of X in (3) can be written in the form $X^{8^i+8^j}$ or X^{8^i}, and so it is clear that (3) $f = \phi^{-1} \circ F \circ \phi$ is a quadratic polynomial map from k^n to k^n. As in the previous example, it is helpful to write $\phi^{-1} : K \longrightarrow k^n$ using matrix notation

$$\mathsf{A}\,\mathsf{X} = \mathsf{x},$$

where $\mathsf{X} = (X^{8^0}, X^{8^1}, \ldots, X^{8^{n-1}})^T$, $\mathsf{x} = (x_0, x_1, \ldots, x_{n-1})^T$, and A is an $n \times n$ matrix with entries from K that can easily be found by writing each X^{8^i} as a polynomial in y with coefficients in $k[x_0, x_1, \ldots, x_{n-1}]$.

The polynomial (3) can be factored easily by a computer algebra system like Magma [CAG05]. Depending on the coefficients $a_1, \ldots, a_8$ of F, and on the value of n, F may have zero, one, or more linear factors in X. Each linear factor $X + \alpha$ with $\alpha \in K$ gives rise to a solution of the corresponding polynomial equations $f_i(x_0, x_1, \ldots, x_{n-1}) = 0$, $i = 0, 1, \ldots, n-1$. Finding the corresponding solutions directly with the help of a good Gröbner bases program such as Faugère's F_4 version in Magma [Fau99] requires exponential time with increasing n as seen in Figure 1.

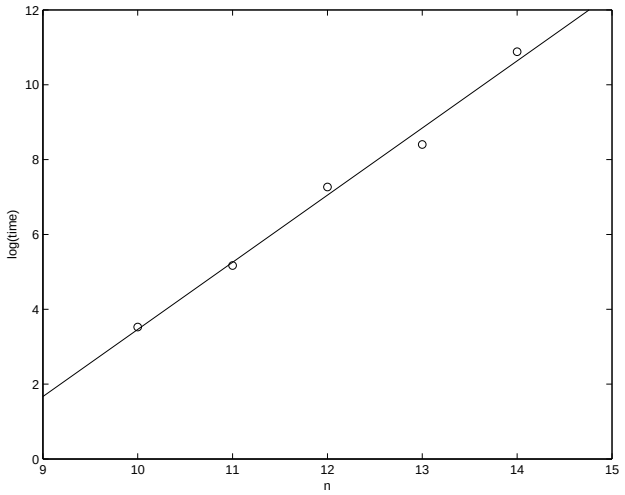


Fig. 1. Computing time for finding a Gröbner basis for n quadratic polynomials in n variables

The quadratic polynomials components of f have too many terms to be displayed here. It could be said that this is an unfair comparison, since we are solving $F(X) = 0$ directly and forcing Faugère's algorithm F_4 to work on a system with a huge number of terms. However, the Gröbner bases algorithm in Magma is very efficient, and removing even a few of the terms in (3) made it much more difficult to find examples where F_4 fails even for large n. It is easy to see why this is to be expected. In principle our algorithm should be better if the degree D of F is fixed due to the complexity estimate $O(D \log q^n)$ for a root finding algorithm [vG03]. On the other hand the complexity of the Gröbner bases algorithm is expected to be exponential in n, the number of variables.

This first non-trivial example shows that the Zhuang-Zi algorithm, even in its most simple form using only Step 1 and the Berlekamp algorithm, sometimes has an advantage over the best Gröbner bases algorithms.

4.3 A Non-Trivial Example

We now give a non-trivial example where $F(X)$ is of very high degree and therefore cannot be solved with the simplest form of the Zhuang-Zi algorithm as discussed in the previous section.

Let $q = 4$, $k = GF(q)$, as in Section 4.1, take $g(y) \in k[y]$ to be the irreducible polynomial

$$g(y) = y^{12} + y^{11} + ay^{10} + ay^9 + y^8 + y^7 + y^5 + a^2y^4 + ay^3 + a^2y^2 + ay + a,$$

and define $K = k[y]/(g(y))$, a degree $n = 12$ extension of k.

Let $F(X) \in K[X]$ be the polynomial

$$\begin{aligned} F(X) =& a^2X^{17664} + X^{5440} + aX^{5376} + X^{4416} + aX^{4096} + aX^{1360} \\ &+ X^{1344} + X^{1280} + a^2X^{1024} + a^2X^{336} + aX^{320} + a^2X^{276} \\ &+ X^{85} + aX^{84} + aX^{64} + aX^{21} + X^{20} + a \end{aligned}$$

It is easy to check that each exponent of X in $F(X)$ is a sum of powers of four, and that the exponent with the most powers of four is $5440 = 4^3 + 4^4 + 4^5 + 4^6$. Therefore the components of $f = \phi^{-1} \circ F \circ \phi$ will be of degree four. As before, there are too many terms in each $f_i(x_0, x_1, \ldots, x_{n-1})$ to be displayed here.

The degree of F prevents us from finding the roots of $F(X) = 0$ directly. Also, the F_4 implementation in Magma failed to find a Gröbner basis for $f_0, f_1, \ldots, f_{n-1}$ due to the fact that memory requirements exceeded the available resources on our PC (1.73 GHz, 1 GB of RAM). However, the Zhuang-Zi algorithm found the polynomial

$$S(X) = X^{276} + aX^{85} + a^2X^{84} + a^2X^{64} + a^2X^{21} + aX^{20} + a$$

and with it the solutions $\{1, a\}$ of $F(X) = 0$.

Other similar examples whose solutions can not be obtained from the Gröbner bases algorithm, and which require several iterations of the full algorithm, can be generated in this way. We omit them here due to space constraints.

5 Discussion

What we propose here is not just a new algorithm, but a new way to look at the problem of solving a set of multivariate polynomial equations over a finite field. We lift the problem to an extension field where it becomes a single variable problem, and then use existing efficient techniques for solving a polynomial equation in a single variable over a finite field. In this way we actually unify the multivariate case and the single variable case. This also means that sometimes it can be beneficial to view a single variable polynomial equation over a given finite field as a set of multivariate polynomial equations over a smaller finite field. We believe that this is an approach that merits further investigation.

Our experiments above have shown that there are cases where the Zhuang-Zi algorithm will succeed in finding a solution to a set of polynomial equations, whereas Gröbner bases algorithms will fail due to space and/or time limitations.

There are many interesting directions for further research. One question is the complexity of the Zhuang-Zi algorithm for a set of n nonlinear equations in n variables. In general the complexity will be exponential in n, though for certain types of equations the Zhuang-Zi algorithm will be much faster. A very good testing ground is the class of equations that arise in connection with the HFE multivariate public key cryptosystem, which we are currently investigating.

Another important task is to make a systematic comparison of the Zhuang-Zi algorithm with other algorithms, for example with the new Gröbner basis algorithms like F_4 and F_5, will give a better understanding of its complexity. The Zhuang-Zi algorithm will not be better in a general way, but rather, we believe that the algorithms can be complimentary to each other.

One interesting point of the Zhuang-Zi algorithm is that it is actually closely related to the XL algorithm. If in Step 3 we do not look for a polynomial of lowest degree but instead we look for a polynomial of the form

$$\sum_{i=0}^{n-1} A_i X^{q^i} + B,$$

then the Zhuang-Zi algorithm is equivalent to the XL algorithm. It would be interesting to consider how to combine the two algorithms together.

If $F(X)$ is known to have a large number of solutions (larger than the threshold for factorization degree D), then the Zhuang-Zi algorithm cannot succeed. One strategy that can be employed is to add randomly chosen polynomials $f_n, \ldots, f_{n'-1}$ to the original set of polynomial $f_0, f_1, \ldots, f_{n-1}$. It is very likely that the resulting new system of equations will have fewer solutions, and that Zhuang-Zi may be successful. If not, we can start over with a new set of randomly chosen polynomials.

A much more general consideration is of the so-called hard cases, those equations that are generically hard to solve using any kind of general algorithm. We believe our approach may provide some insight into how to look at this problem. Among other applications, any result in this direction will have a very strong connection and impact on the provable security of multivariate public key cryptosystems.

There is considerable room to improve and optimize the implementation of the algorithm. For example, the algorithm relies very much on how the big field K is implemented. Also, we expect that it is possible to speed up the Gaussian elimination process by using sparse matrices. If the Zhuang-Zi algorithm is viewed as a philosophy for solving equations, then it is clear that there is great flexibility to create efficient variants in the degree reduction steps, variants tailored to fit with specific efficient polynomial equation solvers and factorization algorithms.

References

[Buc65] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenrings nach einem nulldimensionalen Polynomideal*. Universität Innsbruck, 1965.

[CAG05] University of Sydney Computational Algebra Group. The MAGMA computational algebra system for algebra, number theory and geometry. http://magma. maths.usyd.edu.au/magma/, 2005.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In B. Preenel, editor, *Advances in cryptology, Eurocrypt 2000*, volume 1807 of *LNCS*, pages 392–407. Springer, 2000.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra*, 139:61–88, June 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *International Symposium on Symbolic and Algebraic Computation — ISSAC 2002*, pages 75–83. ACM Press, July 2002.

[GCL92] Keith O. Geddes, Stepen R. Czapor, and George Labahn. *Algorithms for Computer Algebra*. Amsterdam, Netherlands: Kluwer, 1992.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and intractability, A Guide to the theory of NP-completeness*. W.H. Freeman, 1979.

[Knu81] Donald Knuth. *The Art of Computer Programming*. Adison Wesley, 2nd edition, 1981.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In M. Wiener, editor, *Advances in cryptology – Crypto '99*, volume 1666 of *LNCS*, pages 19–30. Springer, 1999.

[LN03] Rudolf Lidl and Harald Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Application*. Cambridge University Press, 2nd edition, 2003.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynomial-tuples for efficient signature verification and message encryption. In C. G. Guenther, editor, *Advances in cryptology – EUROCRYPT '88*, volume 330 of *LNCS*, pages 419–453. Springer, 1988.

[Pat96] Jacques Patarin. Hidden Field Equations (HFE) and Isomorphism of Polynomials (IP): Two new families of asymmetric algorithms. In U. Maurer, editor, *Eurocrypt'96*, volume 1070 of *LNCS*, pages 33–48. Springer, 1996. Extended Version: http://www.minrank.org/hfe.pdf.

[Pat00] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88. *Designs, Codes and Cryptography*, 20:175–209, 2000.

[Smi52] David E. Smith. *History of Mathematics*, volume 1 and 2. New York: Dover, 1951, 52.

[vG03] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2nd edition, 2003.

Appendix: Zhuang Zi

Zhuang Zi, also known as Zhuang Zhou, Chuang Tzu, Chuang Tse, and Chuang Chou (369 BC – 286 BC), was a Chinese philosopher, who lived during the Warring States era. He also lived in a time, which is called a time of the Hundred Schools of Thoughts, during which most major schools of Chinese Philosophy were originated. Zhuang is actually his family name and Zhou the first name. The name Zhuang Zi is used as a way to show respect for him. He categorically belongs to the school of Taoism.

One the most famous stories about him and his philosophy is from his book also known as Zhuang Zi.

> *Once upon a time, I, Zhuang Zhou, dreamt I was a butterfly, fluttering here and there. I was conscious only of my being as a butterfly, unaware that I was Zhou. Soon I awaked, and there I was, myself again. Now I am confused and do not know whether I was then a man dreaming I was a butterfly, or whether I am now a butterfly, dreaming I am a man.*

The first author has proposed this algorithm and he envisions the transition between the vector space on a small finite field and the equivalent large field to be just like the idea of Zhuang Zhou and the butter fly. For this reason he has suggested to name this algorithm after Zhuang Zi.