# Public-Key Cryptography from New Multivariate Quadratic Assumptions

Yun-Ju Huang
Kyushu University
cs@crypto.tw

Feng-Hao Liu
Brown University
fenghao@cs.brown.edu

Bo-Yin Yang
Academia Sinica
by@crypto.tw

May 15, 2012

### Abstract

In this work, we study a new multivariate quadratic (MQ) assumption that can be used to construct public-key encryption schemes. In particular, we research in the following two directions:

- We establish a precise *asymptotic* formulation of a family of hard MQ problems, and provide empirical evidence to confirm the hardness.
- We construct public-key encryption schemes, and prove their security under the hardness assumption of this family. Also, we provide a new *perspective* to look at MQ systems that plays a key role to our design and proof of security.

As a consequence, we construct the *first* public-key encryption scheme that is *provably secure* under the MQ assumption. Moreover, our public-key encryption scheme is efficient in the sense that it only needs a ciphertext length $L + \text{poly}(k)$ to encrypt a message $M \in \{0, 1\}^L$ for any un-prespecified polynomial $L$, where $k$ is the security parameter. This is essentially *optimal* since an additive overhead is the best we can hope for.

## 1 Introduction

Exploring different types of assumptions has been an important direction in the agenda of cryptography research. For robustness, this reduces the risk of a new mathematical/algorithmic/hardware breakthrough that breaks a particular assumption and renders all its following constructions insecure; for versatility, different assumptions usually have advantages for different applications. However, over the past 30 years, only a few candidates of computational problems are built as foundations on which more exciting cryptographic applications can build; for example, some well-structured algebraic, coding, or geometric problems (and their variants): DDH [DH76], Paring (some are instantiated by elliptic curves) [BF01], RSA [RSA78], McEliece [McE78], LWE [AD97, Reg09, Pei09], and some recent works for combinatorial problems [ABW10].

This work is in a step of this agenda. We study a new type of assumption inspired from the field of solving multivariate quadratic (MQ) equations. In particular, we give the first asymptotic formulation of a family of MQ problems that enjoy some good mathematical structures and hardness. Thus one can use this formulation as a base to construct more interesting crypto primitives, such as public-key encryption schemes. Our assumption considers a family of problems that can be viewed as solving MQ equations described as the followings (informally) :

**Definition 1 (The Hard Task (Informal))** *Let $\mathbb{F}_q$ be a finite field, and $H$ be some subset of $\mathbb{F}_q$. Let $S$ be a multivariate quadratic system with $n$ variables and $m$ polynomials whose coefficients are sampled from some distribution $\chi$.*

*Then a solver $A$, given $(S, \vec{y} = S(\vec{x}))$ where $\vec{x}$ is sampled uniformly from $H^n$, is asked to output some $\vec{x}'$ such that $S(\vec{x}') = \vec{y}$.*

Actually, solving systems of non-linear equations is not a new topic, for it has been studied in commutative algebra and algebraic geometry, at least since Francis Sowerby Macaulay [Mac02] (1902). Around the turn of the millennium, these techniques [CKPS00] were also found that they can be used as a cryptanalytic step. Claims (e.g. **XSL** [CP02]) concerning such techniques, today called "algebraic cryptanalysis", were often over-optimistic, but equation-solvers over different finite fields such as **XL** [CKPS00], **F4, F5** [Fau02, FJ03] are now significant topics for crypto.

The fundamental reason that algebraic cryptanalysis is not all-powerful is that solving systems of non-linear equations does not scale well with the parameters even with Moore's Law. Theoretically, solving multivariate non-linear systems, or even just multivariate quadratic (MQ) equations has been proven to be NP-hard [FY80, PG97] in the worst case, and practically, all the proposed solvers fail to solve the systems efficiently (i.e. in polynomial-time) for *most* non-trivial distributions [BGP06, LLY08].

The above approach hints at inherent hardness in solving MQ equations, and consequently MQ could be a good choice as a base for designing crypto systems. Although this direction in fact has been considered for the last 20 years, however, it has had a rocky history. Many schemes were proposed, broken, sometimes patched, and sometimes broken again (see [MI88, Pat95, PGC98, DFS07, DFSS07, DDY$^+$08], and [Pat96, BPS08, CCD$^+$08, BFP11]). One objection frequently voiced is that the security of these systems is often ad-hoc, and thus hard to evaluate. Fundamentally, these approaches mostly were designed with a practical goal in mind. As a result, they considered concrete and fixed-parameter constructions, with a design security of, e.g., $2^{80}$, with specialization to signatures with 160-bit hashes and optimizing for speed. Since MQ was examined not as a hardness basis but only as the most obvious attack or even some sanity check, the designers' mindsets were not focusing on how to construct a reduction for their security proof, nor about extending their schemes in an asymptotic way. Thus, it seems that using the hardness to construct crypto construction remains an interesting open direction.

Berbain, Gilbert, and Patarin [BGP06] explored this and constructed efficient pseudorandom generators (PRGs) based on the hardness of solving MQ equations. Berbain *et al.* considered fixed and concrete-parameter constructions, yet an asymptotic formulation of hard problems is implicit in their work. Consequently, many primitives such as pseudorandom functions (PRFs), symmetric encryptions, etc., in the Minicrypt world (i.e., one way functions exist) [Imp95] can be constructed based on this formulation of hard problems. For the more sophisticated Cryptomania world (i.e., public-key crypto systems exist) [Imp95], the possibilities have not yet been explored in the MQ literature. This line of research will be our main focus in the rest of this paper.

**Our Main Results.** In this work, we study a new MQ assumption that can be used to construct more sophisticated primitives such as public-key encryptions in the Cryptomania world [Imp95]. In particular, we research in the following two directions:

- On the one hand, we establish a precise *asymptotic* formulation of a family of hard problems, and provide empirical evidence to confirm the hardness. Since there are many practical solvers studied and implemented during the studies of algebraic attacks, we use these to examine the hardness of the problems.

- On the other hand, we construct public-key encryption schemes, and prove their security under the hardness assumption of the said family. Also, we provide a new *perspective* to look at MQ systems that plays a key role to our design and proof of security.

As a consequence, we construct the first public-key encryption scheme that is *provably secure* under the MQ assumption. Moreover, our public-key encryption scheme is efficient in the sense that it only needs a ciphertext length $L + \mathrm{poly}(k)$ to encrypt a message $M \in \{0,1\}^L$ for any unprespecified polynomial $L$.[1] This is essentially *optimal* since an additive overhead is the best we can hope for.

The MQ assumption has some interesting properties for its potential. In the following, we will discuss that the MQ problems share some structures with the learning with error (LWE) problems[Reg09, GPV08, PVW08]. Thus the MQ assumption may also enjoys the versatility as LWE. On the other hand, there are many experiences or fast implementations under a variety of hardwares [BGP06, BERW08, CCC$^+$09] in the MQ literature, and thus this can be a good basis for practical applications.

Note: we are unaware of any reductions between our MQ assumption or indeed any MQ-type assumptions and lattice-related ones such as LWE. Furthermore, lattice problems have been studied for a much shorter period of time than equation-solving, and new methods such as BKZ 2.0 [CN11] are still proposed. So it is difficult to compare PKC constructions based on lattice-related hard problems and MQ problems. The comparison is a very interesting research direction but outside the scope of this paper. This paper will simply focus on the MQ assumption and its consequent constructions.

**A Closer Look at Our Assumption.** In the following, we take a closer look at our assumption and techniques, and still maintain a high-level perspective for intuitions. First, we give some notation for convenience of exposition. Let $\mathbb{F}_q$ be a field which we use in the following discussion, and let $S$ describe a multivariate quadratic system with $n$ variables and $m$ polynomials. For example, the following system is one with 3 variables and 2 polynomials, and for a concrete explanation we set $q = 13$.

$$S \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \overset{\mathrm{def}}{=} \begin{cases} x_1x_3 + x_2^2 + 3x_1 + 2 \\ x_1x_2 + 2x_1 + 2x_2 + 7 \end{cases} \tag{1}$$

In addition to viewing $S$ as a set of polynomials, we can view the above system $S$ as a function mapping from $\mathbb{F}_q^3$ to $\mathbb{F}_q^2$. For example, $S([1,2,3]^T) = [12,2]^T$, where $T$ denotes transposes of vectors. In the rest of the paper, we use $S[\cdot]$ to denote a system of polynomials, and $S(\cdot)$ to denote the corresponding function. Now we are ready to describe the hard problem of our assumption with more details (still informally). Note that here the system $S$ includes quadratic terms, linear terms and constant terms. Throughout the paper, we will use $S$ to denote a system with all quadratic, linear and constant terms.

[1] $k$ is the security parameter.

**Definition 2 (The Hard Task (Informal))** *Let $q$ be a large enough prime, and $H$ be some small subset of $\mathbb{F}_q$. Let $S$ be a multivariate quadratic system with $n$ variables and $m = \Theta(n)$ polynomials sampled from a distribution where the coefficients of linear and constant terms are uniformly random, and the quadratic terms come from independent Gaussian distributions with means $0$ and moderately large standard deviations.*

*Then a solver $A$, given $(S, \vec{y} = S(\vec{x}))$ where $\vec{x}$ is sampled uniformly from $H^n$, is asked to output some $\vec{x}'$ such that $S(\vec{x}') = \vec{y}$.*

To make the seemingly intimidating parameters more reader-friendly, we give an intuitive-level discussion as follows. First, we observe that depending on the parameters, solving MQ equations can be easy or hard. As discussed in [BGP06], when $m$ is significantly larger or smaller than $n$, solving the problem is easy. The interesting hard instances fall on the cases when $m$ is close to $n$, as stated in the above definition that $m = \Theta(n)$. Moreover, the problem is believed to be not only hard in the worst case, but hard on average over random instance of $S$, and random input $\vec{x}$. Under a series of empirical studies and theoretical studies [AFI$^+$04, Die04, YC04, YCBC07] for the best known solvers, the best known algorithms still remain exponential-time.

Previously, [LLY08] observed (from experiments) that even if the instance $S$ is drawn from a biased distribution (whose quadratic coefficients are not uniform but instead sparse), solving the problem is still hard. This result hints at an intuition that MQ problems are hard for most (non-trivial) distributions from which $S$ is drawn. In this work, we further test this intuition by investigating the case that the instance $S$ is drawn from a distribution whose quadratic coefficients come from Gaussian distributions with moderately large standard deviation, and the input $\vec{x}$ is drawn from a smaller subset $H^n$. Our experiment results in Section 6 confirm our intuition that the problem does not become significantly easier. In the following paragraphs, we explain how and why this type of assumption and hardness help our design.

We remark that here we only give a structural description of the problem, and leave the precise quantitative statement in Section 3. Before going to the detailed calculation of numbers, we first focus on the structural properties of the hard problem and maintain a high-level perspective.

**Overview of Our Construction.** Inspired by the recent constructions of public-key crypto systems by learning with error (LWE) problems [Reg09], we observe that the problem in Definition 2 also shares the same structure with LWE. We can take advantage of this similarity for our construction of public-key encryption schemes. This is a new perspective of how we can view MQ equations.

First, let us take a look at the LWE problem, which can be stated as the following: let $A \in \mathbb{F}_q^{m \times n}$ be a matrix, and $\vec{b}$ be a vector $\vec{b} = A \cdot \vec{s} + \vec{e}$, where $\vec{s} \in \mathbb{F}_q^n$ is some secret, and $\vec{e}$ comes from some error distribution. The task of the LWE problem is to find out $\vec{s}$ given a random $A$, and an induced $\vec{b}$.

We highlight the similarity by way of the following observation: recall that the task of the problem in Definition 2 is to invert $\vec{y} = S(\vec{x})$ given $S, \vec{y}$. We can rewrite $\vec{y}$ into $S(\vec{x}) = L \cdot \vec{x} + \vec{d} + R(\vec{x})$, where $L$ is the matrix of the terms of linear coefficients, $\vec{d}$ is the coefficient vector of constant terms, and $R(\vec{x})$ are the mapping by the quadratic terms. Take Equation 1 for example, we can rewrite the expression of $S(\vec{x})$ as:

$$S(\vec{x}) = \left\{ \begin{array}{c} x_1x_3 + x_2^2 \\ x_1x_2 \end{array} + \begin{array}{c} 3x_1 \\ 2x_1 + 2x_2 \end{array} + \begin{array}{c} 2 \\ 7 \end{array} \right. = R(\vec{x}) + \begin{pmatrix} 3 & 0 & 0 \\ 2 & 2 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 2 \\ 7 \end{pmatrix}$$

In this expression, $S(\vec{x})$ is a combination of an affine transformation $(L \cdot \vec{x} + \vec{d})$ plus some quadratic mapping $R(\vec{x})$. We remark that without loss of generality, we can assume $\vec{d} = 0$, since solving a multivariate system with all 0s for the constant coefficients is equivalent to solving that with random constant coefficients.[2] Then if we view the quadratic terms as *noise* (analogous to the vector $\vec{e}$), the shared structure becomes apparent. Thus, the ideas that com from using LWE may be translated into candidates of constructions by MQ problems.

However, to bridge the two problems, we need to deal with some subtleties. In the LWE problems, the noise (error vector $\vec{e}$) comes from a Gaussian distribution that has "moderately" large standard deviation. Intuitively, if the standard deviation is too small, then the problems become easier; on the other hand, if it is too large, then the ciphertexts (constructed from LWE) become undecryptable. Thus, in this series or works [Reg09, GPV08, PVW08], certain ranges of parameters for stds have been identified such that both the hardness of the problems and the correctness of the decryption hold simultaneously.

When MQ problems are viewed in this way, we also need to argue that the noise $R(\vec{x})$ is also "moderate." To achieve this, we use the structure of the assumption that the coefficients of each quadratic term come from Gaussian distributions with moderately large standard deviations, and the input $\vec{x}$ comes from a small subset $H^n \subseteq \mathbb{F}_q^n$. That property allows us to bound the size of the noise $R(\vec{x})$. On the other hand, we need to examine the hardness of the problem for these parameters. To do so, we conduct experiments under what to our knowledge the best quadratic equation solver. Our experiment results confirm our intuitions that MQ problems do not become significantly easier under any (non-trivial) particular distribution of the inputs $S$ and $\vec{x}$. This particularly gives us evidence of the hardness of the problem in Definition 2, which we can use to construct public-key encryptions.

**Our First Construction of Encryption for Bits.** In our first attempt, we construct a public-key encryption scheme for bits. This construction is similar in spirit to those LWE-based constructions [Reg09, GPV08, PVW08]. Because of the similarity, here we omit discussions of intuitions and refer the curious readers to [Reg09, GPV08, PVW08]. For confirmations of the correctness and security, we present a formal description of the scheme and a formal security proof in Section 3 and 3.3. Here we give an informal outline of the construction:

- In key generation, the algorithm samples an MQ system $S$ with $n$ variables and $m = c \cdot n$ polynomials, and $\vec{x} \in H^n$. Then it sets the public key to be $(S, \vec{y} = S(\vec{x}))$, and the secret key to be $\vec{x}$.

- To encrypt a bit $b$, the encryption algorithm samples $\vec{r} \in H^m$, and computes $(c_1, c_2) = (\vec{r}^T \cdot L,\ \vec{r}^T \cdot (\vec{y} - \vec{d}) + b \cdot [q/2])$. Recall that $L$ is an $m \times n$ matrix, and $m > n$. Thus, given $\vec{r}^T \cdot L$, $\vec{r}$ is still hidden information theoretically.

[2] There is a simple reduction showing that solving $(S, \vec{y} = S(\vec{x}))$ for $S$ contains random constant coefficients is equivalent to solving $(S', \vec{y}' = S'(\vec{x}))$, where $S'$ has the same distribution as $S$, except for the 0 constant coefficients.

- To decrypt, the algorithm computes $t = c_2 - c_1^T \cdot \vec{x}$. It outputs 1 if and only if $|t - q/2| \leq q/4$.

**Security Proof.** The key to the security proof of the bit-encryption scheme is based on a proof that relates the hardness of the assumption to some pseudorandom distribution. Namely, suppose the problem in Definition 2 is hard, then $(S, S(\vec{x}))$ is indistinguishable from $(S, U_m)$ where $U_m$ is uniform over $\mathbb{F}_q^m$. Moreover, we prove a more general theorem that suppose there exist a distribution over the quadratic terms of $S$, and a subset $H \subseteq \mathbb{F}_q$ such that the problem is hard, then $(S, S(\vec{x}))$ is indistinguishable from $(S, U_m)$. The crux of our proof is a new application of the new version of Goldreich-Levin Theorem by Dodis *et. el* [DGK$^+$10].

We remark that this general theorem also, as a consequence, implies Theorem 2 plus 3 in [BGP06], and Proposition 5 plus 6 in [LLY08] as its special cases.[3]

**Improving Efficiency Using KEM.** Feasibility results for bit-encryptions are nice but not quite satisfactory. One general technique to improve efficiency is to use *Key Encapsulation Mechanism* (KEM). We know that to use KEM, it is sufficient to have an efficient symmetric encryption scheme or a pseudorandom generator (PRG). (Note that a pseudorandom generator implies an efficient symmetric encryption scheme.) Although there are many implementations of PRGs and thus symmetric encryptions as well[BM84, BBS86, Kal86, FSS07, HILL99, Hol06, HRV10], the constructions are either not practically efficient, or require some additional assumption(s).

Here we further observe that the MQ assumption (Definition 2) already gives us an efficient construction of a certain form of PRG[4] that is sufficient to implement the KEM technique. As a consequence, in the resulting scheme, we are able to achieve a public-key encryption scheme that only needs a ciphertext length $L + \text{poly}(k)$ to encrypt a message $M \in \{0,1\}^L$ for any un-prespecified polynomial $L$, where $k$ is the security parameter. This is essentially *asymptotically optimal* since we know the ciphertext length must be at least as large as the message (otherwise there will be decryption errors), and an additive overhead in the security parameter is the the best we can hope for.

# 2 Preliminary

## 2.1 Notation

All vectors are assumed to be column vectors. Unless stated otherwise, all scalar and vector operations are performed modulo $q$. We use arrow notation to represent a vector, and subscripts to represent the corresponding element, i.e. $\vec{r} \in \mathbb{F}_q^n$ means $\vec{r}$ is a vector of $n$ elements in $\mathbb{F}_q$ and $\vec{r}_i$ means the $i$-th element of the vector. We denote the transpose of a vector $\vec{r}$ as $\vec{r}^T$.

For simplicity we will assume that $q$ is an odd prime. We represent elements in $\mathbb{F}_q$ by integers within the range $[-(q-1)/2, (q-1)/2]$. We denote the inner product of $\vec{a}$ and $\vec{b}$ as $\langle \vec{a}, \vec{b} \rangle$, or $\vec{a}^T \cdot \vec{b}$.

**Definition 3 (Computational Indistinguishability and Pseudorandom Distributions)** *Let $k$ be the security parameter and $\{X_k\}, \{Y_k\}$ be ensembles, where $X_k$' s and $Y_k$'s are probability distributions over $\Omega^{\ell(k)}$ for some polynomial-lengthed finite set $\Omega$ for some polynomials $\ell$. We say that*

[3] We present our theorem and assumption in asymptotic forms, and both [BGP06, LLY08] presented their theorems in concrete parameters.

[4] The PRG constructed by the MQ assumption is somewhat non-standard but is sufficient for KEM. See Section 5 for further discussions.

*$\{X_k\}, \{Y_k\}$ are computationally indistinguishable if for all* PPT *(or non-uniform polynomial size circuit) distinguisher $A$, there exists some negligible function* $\mathsf{ngl}(\cdot)$ *such that*

$$|\Pr[A(X_k) = 1] - \Pr[A(Y_k) = 1]| < \mathsf{ngl}(k).$$

*We use $\{X_k\} \approx_c \{Y_k\}$ to denote computational indistinguishability for simplicity.*

*If $Y_k$'s are the uniform distributions over $\Omega^{\ell(k)}$, and $\{X_k\} \approx_c \{Y_k\}$, then we say $X_k$'s are* pseudorandom *distributions.*

## 2.2 Multivariate Quadratic Problems

**Definition 4 (Multivariate Quadratic Polynomials)** *Let $q \in \mathbb{N}$ be a prime and $\mathbb{F}_q$ be the finite field with order $q$. Let $n \in \mathbb{N}$ be parameters. A quadratic multivariate polynomial with $n$ variables can be denoted as the following form:*

$$Q[\vec{x}] = \sum_{1 \le j \le k \le n} \alpha_{j,k} x_j x_k + \sum_{1 \le j \le n} \beta_j x_j + \gamma,$$

*where $\vec{x} = (x_1, x_2, \ldots, x_n)$ is a vector of the $n$ variables, and $\alpha_{j,k}, \beta_j, \gamma \in \mathbb{F}_q$ are the coefficients to the corresponding monomials.*

**Definition 5 (Multivariate Quadratic Systems)** *A multivariate quadratic system is a set of multivariate quadratic polynomials. More precisely, let $q \in \mathbb{N}$ be a prime, $n, m \in \mathbb{N}$ be parameters, and let $\vec{x}$ be a vector of the $n$ variables. A quadratic multivariate system with $n$ variables and $m$ polynomials can be written as $S = (Q_1, Q_2, \ldots, Q_m)$ where each $Q_i[\vec{x}]$ is a multivariate quadratic polynomial of the above form.*

*For compactness of the notation, we write the coefficients in a matrix form as the following. Let $R \in \mathbb{F}_q^{m \times n \times n}$ be $m$ $n \times n$ matrices, $L \in \mathbb{F}_q^{m \times n}$ be a $m \times n$ matrix, and $\vec{d} \in \mathbb{F}_q^n$ be a vector. For each polynomial $Q_i$, let $R_{i,j,k}$ be the coefficient of the monomial $x_j x_k$*[5]*, $L_{i,j}$ be that of $x_j$, and $\vec{d_i}$ be the coefficient for constant terms.*

*Using this notation, we can write each polynomial as:*

$$Q_i[\vec{x}] = \vec{x}^T \cdot R_i \cdot \vec{x} + L_i \cdot x + \vec{d_i},$$

*where $R_i$ is the $i$-th $n \times n$ matrix, and $L_i$ be the $i$-th row of $L$. Then we denote the system by:*

$$S[\vec{x}] = R[\vec{x}] + L[\vec{x}] + \vec{d},$$

*where $R[\vec{x}]$ is a collection of the quadratic part for the polynomials.*

*To specify a system with $n$ variables and $m$ polynomials, we denote $S = (R, L, \vec{d})$ by the matrix representation as above.*

**Definition 6 (Multivariate Quadratic Systems Evaluated at Points as Functions)** *Let $q \in \mathbb{N}$ be a prime, and $S = (R, L, \vec{d})$ be a quadratic system with $n$ variables and $m$ polynomials as above, denoted as $S[\vec{x}]$. Let $\vec{z} \in \mathbb{F}_q^n$ be a vector. We write $\vec{y} := S(\vec{z})$, a vector in $\mathbb{F}_q^m$ where we evaluate $\vec{y}$ by plugging each $x_i$ with the value $z_i$. In this case, $S(\cdot)$ is viewed as a function $S : \mathbb{F}_q^n \to \mathbb{F}_q^m$.*

[5] For symmetry, we define $R_{i,j,k}$ to be 0 for $k < j$.

*We use $[\cdot]$ to denote polynomials, and $(\cdot)$ to denote functions, respectively. In particular, we use $S[\cdot], R[\cdot], L[\cdot]$ to denote the systems of polynomials , and $S(\cdot), R(\cdot), L(\cdot)$ to denote the corresponding functions.*

**Definition 7 (Multivariate Quadratic Problems)** *Let $n, m, q \in \mathbb{N}$ be parameters such that $q$ is a prime, let $\chi$ be a distribution between $\mathbb{F}_q^{m\times n\times n}$, and let $H \subseteq \mathbb{F}_q$. The goal for a solver $A$ to the (average-case) multivariate quadratic problem $MQ(n, m, q, \chi, H)$ is that $A$ on a random instance $(S, S(\vec{x}))$ tries to output some $\vec{x}' \in \mathbb{F}_q^n$ such that $S(\vec{x}') = S(\vec{x})$, where $S = (R, L, \vec{d})$ with $R \leftarrow \chi$, $L \leftarrow \mathbb{F}_q^{m\times n}$, $\vec{d} \leftarrow \mathbb{F}_q^m$, and $\vec{x} \leftarrow H^n$. If $A$ does so, we say it successfully solves the instance.*

**Definition 8 (Hardness of a MQ Family)** *Let $k$ be the security parameter, $n, m, q : \mathbb{N} \to \mathbb{N}$ be efficiently computable and polynomially bounded such that $q$ is an odd prime. Let $\chi$ be a distribution over $\mathbb{F}_q^{m\times n\times n}$ and $H \subseteq \mathbb{F}_q$. We say that the family $MQ(n, m, q, \chi, H)$ is hard to solve if for every* PPT *solver $A$, there exists some negligible function $\mathsf{ngl}(\cdot)$ such that the following holds for all sufficiently large $k$:*

$$\Pr_{\substack{S \leftarrow MQ(n,m,q,\chi,H)\\ \vec{x} \leftarrow H^n}} \left[\vec{x}' \leftarrow A(S, S(\vec{x})) : S(\vec{x}') = S(\vec{x})\right] < \mathsf{ngl}(k).$$

# 3 Public-Key Encryption Schemes For Bits

In this section, we show a construction of public-key encryption schemes (for bits) under the hardness of some specialized MQ problem. We present our results in the following order: (1) the hardness assumption, (2) the construction of the scheme, and (3) the analysis.

## 3.1 The Assumption

**Definition 9 (MQ Hardness Assumption)** *Let $k$ be the security parameter. For every constant $c > 1 \in \mathbb{N}$, every efficiently computable and polynomially bounded $n, m, q : \mathbb{N} \to \mathbb{N}$, $\alpha : \mathbb{N} \to [-q/2, q/2]$ and every $0 < \beta \leq [q/2]$ such that (1) $m = cn$, (2) $q$ is prime, (3) $\alpha = O(1)$, let $\Phi_\alpha$ be the distribution of $m \times n \times n$ identical independent discrete Gaussian distribution $D_\alpha$'s with mean $0$, standard deviation $\alpha$, namely, each $D_\alpha$ samples $z \leftarrow N(0, \alpha^2)$ (normal distribution with mean $0$, and standard deviation $\alpha$), and then outputs $\lfloor z \rceil$ (mod $q$), and let $H_\beta = \{-\beta, -\beta + 1 \ldots, \beta - 1, \beta\}$.*

*Then the problem $MQ(n, m, q, \Phi_\alpha, H_\beta)$ is hard to solve.*

As discussed in the introduction, we need to choose the parameters $\alpha$ such that $|R(\vec{x})|$ is "moderate" for two aspects. First, $\alpha$ cannot be too large, otherwise there will be decryption errors. On the other hand, if $\alpha$ is too small, then with high probability, most coefficients are 0, so the system becomes sparse. There are known attacks for sparse systems where there are only $o(1)$ non-zero coefficients, so in our assumption, the $\alpha$ cannot fall into this region. In our setting, $\alpha = O(1)$ implies that each quadratic terms has at least a constant probability not being zero, and thus there will be $O(n^2)$ quadratic terms in expectation. In section 6, we will discuss more details about the parameters and how they influence the hardness of the problem.

**Remark 10** As we discussed in the introduction, the MQ assumption has a similar structure to the LWE assumption. Here we do a brief comparison of the two assumptions for different range of parameters.

For $q$ being superpolynomial, we can show that an MQ instance $(S, S(\vec{x}))$ can be transformed to $(L, b)$ that is statistically close to an LWE instance. The transformation just sets $L$ as the linear part of $S$, and sets $b = S(\vec{x}) + \vec{e}'$, where each coordinate of $\vec{e}'$ comes from some i.i.d. Gaussian with a small std. For $q = \mathsf{superpoly}(k)$, one can show that $b$ is statistically close to $L \cdot \vec{x} + \vec{e}''$ where each coordinate of $\vec{e}''$ comes from i.i.d. Gaussian with a slightly bigger std. Thus, $(L, b)$ is statistically close to an LWE instance, and consequently, there is a simple reduction from MQ to LWE.

In this paper, we need $q = \text{poly}(k)$ for our construction. For this range of parameters, the above argument does not work. In fact, an MQ instance and an LWE instance can be statistically far. Thus, a straitforward reduction from MQ to LWE does not work. We are not aware of any other reduction from any one to the other, and leave this issue as an interesting open question.

Under the above assumption, we are able to obtain the following lemma, which is a key to the security proof of our construction of public-key encryption scheme. In the following section, we are going to prove a more general result as Theorem 22, which directly implies this lemma. Thus, we only put the statement of the lemma.

**Lemma 11** *Let $k$ be the security parameter, and assuming $MQ(n, m, q, \Phi_\alpha, H_\beta)$ be the hard problem as stated in Definition 9. Then $(S, S(\vec{x}))$ is computationally indistinguishable from $(S, U_m)$, where $S \leftarrow MQ(n, m, q, \Phi_\alpha, H_\beta)$, $\vec{x} \leftarrow H_\beta^n$, $U_m$ is the uniform distribution over $\mathbb{F}_q^m$.*

Here we remark that the MQ hardness assumption in Definition 9 can be generalized in the following sense.

**Remark 12** Actually all we need for our construction is to bound the quantity $R(\vec{x})$ (see Section 3.4 for further discussions). Thus any distribution of $S$, and $\vec{x}$ that has the following properties (1) the problem of equation solving is hard, and (2) we are able to bound $R(\vec{x})$, are sufficient for us to construct public-key encryptions. Here for concreteness, we present study $\Phi_\alpha$ and $H_\beta^n$ as a candidate for the hard problem.

## 3.2 Construction of A Public-Key Encryption Scheme for Bits

In this section we present our construction of a public-key bit-encryption scheme.

**Construction of the Scheme $\mathcal{E} = (\text{KeyGen}(\cdot), \text{Enc}(\cdot), \text{Dec}(\cdot))$:**

- $\text{KeyGen}(1^k)$: choose public parameters $n, m, q, \alpha, \beta$, and $\lambda \in \mathbb{N}$ satisfying the following constraints:

  1. $k \cdot \alpha \cdot n^{(2+\lambda)} \cdot m \cdot \beta^2 \leq q/4$.
  2. $m \cdot \log(2n^\lambda + 1) \geq (n+1) \cdot \log q + 2k$.
  3. $n, m, q, \alpha, \beta$ satisfy the condition in the MQ assumption such that $MQ(n, m, q, \Phi_\alpha, H_\beta)$ is hard to solve.

  Then it samples a random instance $(S, S(\vec{x})) \leftarrow MQ(n, m, q, \Phi_\alpha, H_\beta)$, and deontes $\vec{y} = S(\vec{x})$. Then it sets $\mathsf{pk} = (S, \vec{y}) = ((R, L, d), \vec{y})$, $\mathsf{sk} = \vec{x}$.

- $\text{Enc}(b)$ for $b \in \{0, 1\}$: sample $\vec{r} \in H_{n^\lambda}^m$, and outputs $(c_1, c_2) = (\vec{r}^T \cdot L,\ \vec{r}^T \cdot (\vec{y} - \vec{d}) + b \cdot [q/2])$.

- Dec$(c_1, c_2)$: compute $t = c_2 - c_1^T \cdot \vec{x}$. If $|t - q/2| \leq q/4$ then output 1, otherwise 0.

The constraints in the KeyGen capture the two conditions that we need for the encryption scheme. The first condition guarantees the error (from the quadratic terms) won't be too big, and the second condition guarantees the security of the encryption. For a ciphertext $(c_1, c_2) = (\vec{r}^T \cdot L, \vec{r}^T \cdot (\vec{y} - \vec{d}) + b \cdot [q/2]))$, $c_1$ contains a vector of $n$ field elements, which reveal $n \cdot \log q$ bits of information about $\vec{r}$. The second condition says $\vec{r}$ has $m \cdot \log(2n^\lambda + 1)$ bits of entropy, so given $c_1$, $\vec{r}$ still has enough entropy. Then we can apply the leftover hash lemma to argue that $c_2$ is pseudorandom. We remark that the conditions are not hard to satisfy, and in section 5.4 we give concrete parameters that satisfy those constraints, and then analyze the resulting constructions.

**Theorem 13** *Assume the MQ assumption holds for the above parameters. Then the scheme $\mathcal{E}$ is a semantically secure encryption scheme.*

To prove the theorem, we argue its security and correctness in the following two sections.

### 3.3 Proof of Security

To show the scheme is semantically secure, we show that it has the property of message indistinguishability. That is, for the two different messages $0, 1$, we have Enc(0) computationally indistinguishable from Enc(1). The proof uses the property $(S, S(\vec{x})) \approx_c (S, U_m)$ by Lemma 11 in an essential way. In particular, we establish the following lemma.

**Lemma 14** *Let $k, n, m, q, \Phi_\alpha, H_\beta, \lambda$ be parameters that satisfy the constraints above in the* KeyGen. *Then $D_1 = (S, \vec{y}, \vec{r}^T \cdot L, \vec{r}^T \cdot (\vec{y} - \vec{d}))$ is indistinguishable from $D_2 = (S, \vec{y}, \vec{r}^T \cdot L, \vec{r}^T \cdot (\vec{y} - \vec{d}) + [q/2])$.*

**Proof.** To show this lemma, we use the following two intermediate distributions to bridge $D_1$ and $D_2$; namely, $D_1' = (S, \vec{u}, \vec{r}^T \cdot L, \vec{r}^T \cdot (\vec{u} - \vec{d}))$, and $D_2' = (S, \vec{u}, \vec{r}^T \cdot L, \vec{r}^T \cdot (\vec{u} - \vec{d}) + [q/2])$, where $\vec{u}$ is uniform over $\mathbb{F}_q^m$. In particular, we show that $D_1 \approx_c D_1'$, and $D_2 \approx_c D_2'$, and $D_1' \approx_s D_2'$. Thus, we know $D_1 \approx_c D_2$. (Note: $\approx_c$ denotes computational indistinguishability, and $\approx_s$ denotes statistical closeness.)

We establish the following claims:

**Claim 15** *The distribution $D_1$ is computationally indistinguishable from $D_1'$.*

> **Proof of claim:** We prove the claim by contradiction. From Lemma 11, we know $(S, \vec{y}) \approx_c (S, \vec{u})$. Suppose there exists an PPT adversary $A$ that distinguishes $D_1$ and $D_1'$ with noticeable advantage $\varepsilon = 1/\text{poly}(k)$. Then we want to define an adversary $B$ that distinguishes $(S, \vec{y})$ and $(S, \vec{u})$, which leads to a contradiction.
>
> Define $B$ does the following: on input $(S, \vec{z})$ where $\vec{z}$ comes from either $\vec{y}$ or $\vec{u}$, $B$ samples $\vec{r} \in H_{n^\lambda}^m$ and outputs $A(S, \vec{z}, \vec{r}^T \cdot L, \vec{r}^T \cdot (\vec{z} - \vec{d}))$. For the case when $\vec{z}$ comes from $\vec{y}$, $A$'s input distribution is exactly $D_1$, and for the other case $D_1'$. Thus the advantage of $A$ carries directly to $B$. Clearly the running time of $B$ is also in poly$(k)$, and this completes the proof. □

**Claim 16** *The distribution $D_2$ is computationally indistinguishable from $D_2'$.*

> **Proof of claim:** This follows from exactly the same argument as above. □

**Claim 17** *The distribution $\Delta(D_1', D_2') \leq 2^{-k}$, where $\Delta()$ denotes statistical differences.*

**Proof of claim:** To show this claim, we are going to apply the Leftorver Hash Lemma [Vad11] of the following form:

**Lemma 18 (Leftover Hash Lemma[Vad11])** *Let $m, n, k, q$ be parameters such that $m \cdot \log(2n^\lambda + 1) \geq (n+1) \cdot \log q + 2k$, then $(\vec{u}, \vec{r}^T \cdot L, \vec{r}^T \cdot \vec{u})$ is statistically close to $(\vec{u}, \vec{r}^T \cdot L, z)$ by a distance at most $2^{-k}$, where $z$ is a random element in $\mathbb{F}_q$.*

By applying this lemma, we know that $D_1' = (S, \vec{u}, \vec{r}^T \cdot L, \vec{r}^T \cdot \vec{u}) \approx_s (S, \vec{u}, \vec{r}^T \cdot L, z)$, and $D_2' = (S, \vec{u}, \vec{r}^T \cdot L, \vec{r}^T \cdot \vec{u} + [q/4]) \approx_s (S, \vec{u}, \vec{r}^T \cdot L, z)$. This proves the claim. □

■

## 3.4 Proof of Correctness

To show correctness of the encryption scheme, we show that with overwhelming probability over the randomness in the key generation, decrypting a ciphertext recovers the original message.

To achieve this task, first we recall that the ciphertext of $b \in \{0,1\}$ is $(c_1, c_2) = (\vec{r}^T \cdot L,\ \vec{r}^T \cdot (\vec{y} - \vec{d}) + b \cdot [q/2])$. Then we observe that the decryption procedure can be written as:

$$\begin{aligned} t \overset{\text{def}}{=} c_2 - c_1^T \cdot \vec{x} & \\ = \quad & \vec{r}^T \cdot (S(\vec{x}) - \vec{d}) - \vec{r}^T \cdot L \cdot x \\ = \quad & \vec{r}^T \cdot (R(\vec{x}) + L \cdot \vec{x}) + b \cdot [q/2] - \vec{r}^T \cdot L \cdot x \\ = \quad & \vec{r}^T \cdot R(\vec{x}) + b \cdot [q/2]. \end{aligned}$$

The decryption algorithm outputs 1 if $|t - q/2| \leq q/4$, and otherwise 0. Thus if $|\vec{r}^T \cdot R(\vec{x})| < q/4$, the decryption algorithm will correctly recover the bit $b$ originally encrypted. In the following, we bound the probability of $|\vec{r}^T \cdot R(\vec{x})| \geq q/4$.

Before going to the proof, we first look at the high-level discussion as follows: recall that in the setting of the parameters, all the coefficients of $R$ come from $D_\alpha$, i.e., discrete Gaussian distributions with mean 0 and std $\alpha$. Since $\vec{x} \in H_\beta^n$, we know each quadratic monomial $\vec{x}_i \cdot \vec{x}_j$, as a function, can contribute at most $\beta^2$. There are $n^2$ quadratic monomials and each coefficient comes from the Gaussian distributions. Thus the probability that a single polynomial is greater than $k \cdot [n^2 \cdot \beta^2 \cdot \alpha]$ is exponentially small in $k$ by the property that (1) the sum of $n^2$ Gaussians with std $\beta^2 \cdot \alpha$ is still a Gaussian with std $n^2 \cdot \beta^2 \cdot \alpha$, and (2) $\Pr[N(0,\gamma) > k\gamma]$ decreases exponentially with $k$ for any std $\gamma$.

On the other hand, we know that $\vec{r} \in H_{n^\lambda}^m$ and $\vec{r}^T \cdot R(\vec{x})$ can be viewed as $\sum_{i \in [m]} \vec{r}_i \cdot R_i(\vec{x})$. This term can be bounded by $\sum_{i \in [m]} n^\lambda \cdot R_i(\vec{x})$, and in the above paragraph, we have already bounded the probability that each $R_i(\vec{x})$ is too large. Thus, putting these together with the relations of $\alpha, \beta, q$ in the premise, we know the probability $|\vec{r}^T \cdot R(\vec{x})| \geq q/4$ is negligibly small. We present the following lemma that formalizes the above discussions.

**Lemma 19** *Let $n, m, q, \Phi_\alpha, H_\beta$ be the parameters as in the scheme. Then for any $\vec{x} \in H_\beta^n, \vec{r} \in H_{n^\lambda}^m$*

$$\Pr_{S:=(Q,L,\vec{d}) \leftarrow MQ(n,m,q,\Phi_\alpha,H_\beta)} \left[ |\vec{r}^T \cdot R(\vec{x})| \geq q/4 \right] < \mathsf{ngl}(k).$$

**Proof.** [**Liu's Note: Maybe add more intuitions about the proof, english description and what the parameters mean.**] To prove the lemma, first we observe that in our setting of parameters in the KeyGen algorithm, we have $k \cdot \alpha \cdot n^6 \cdot m \cdot \beta^2 \leq q/4$. Thus we have $\Pr[|\vec{r}^T \cdot R_i(\vec{x})| > q/4] \leq \Pr[|\vec{r}^T \cdot R_i(\vec{x})| > k \cdot n^2 \cdot \alpha\beta^2 \cdot m \cdot n^4]$ by simply reordering the terms. In the following we are going to bound the probability of such term by the two claims.

**Claim 20** *Let $\vec{x} \in H_\beta^n$ be any arbitrary, and $R \leftarrow \Phi_\alpha \in \mathbb{F}_q^{m\times n\times n}$. Then for each quadratic polynomial $R_i$,*

$$\Pr[|R_i(\vec{x})| > k \cdot n^2\beta^2 \cdot \alpha] < \mathsf{ngl}(k)$$

**Proof of claim:** Let $\vec{z} = (\beta, \beta, \ldots, \beta) \in \mathbb{F}_q^n$. Since $\vec{x} \in H_\beta$, we know that $|R_i(\vec{x})| \leq |R_i(\vec{z})|$, so in order to bound the probability as stated above, we only need to bound the probability that $|R_i(\vec{z})| > k \cdot n^2\beta^2 \cdot \alpha$ . By the definition of $\Phi_\alpha$, $R_i(\vec{z})$ is the sum of $n^2$ independent Gaussian distributions with mean 0 and standard deviation $\alpha$. Thus $|R_i(\vec{z})| = n^2 \cdot \beta^2 \cdot |N(0,\alpha^2)| = |N(0,(n^2\beta^2)^2 \cdot \alpha^2)|$. Denote $n^2\beta^2 \cdot \alpha$ as $\gamma$

Then by Chernoff bound of the tail probability, we have

$$\Pr[|R_i(\vec{x})| > k \cdot \gamma] \leq \Pr[|R_i(\vec{z})| > k \cdot \gamma] = \Pr[|N(0,\gamma^2)| > k \cdot \gamma] < 2^{-\Omega(k^2)} = \mathsf{ngl}(k).$$

□

**Claim 21** *Let $\vec{r} \in H_{n^\lambda}^m, \vec{x} \in H_\beta^n$ be any two arbitrary vectors, and $R \leftarrow \Phi_\alpha \in \mathbb{F}_q^{m\times n\times n}$. Then*

$$\Pr[|\vec{r} \cdot R(\vec{x})| > k \cdot m \cdot n^{2+\lambda}\beta^2 \cdot \alpha] < \mathsf{ngl}(k).$$

**Proof of claim:** Let $\vec{z} = (n^\lambda, n^\lambda, \ldots, n^\lambda) \in \mathbb{F}_q^m$. We observe that $|\vec{r} \cdot R(\vec{x})| \leq |\vec{z} \cdot R(\vec{x})|$. Thus to bound the desired probability, we only need to bound the probability that $|\vec{z} \cdot R(\vec{x})| > k \cdot m \cdot n^{2+\lambda}\beta^2 \cdot \alpha = k \cdot m \cdot n^\lambda \cdot n^2\beta^2 \cdot \alpha$.

Let $\gamma \overset{\text{def}}{=} n^2\beta^2 \cdot \alpha$. By the above claim , we know that for each $i \in [m]$, $\Pr[|R_i(\vec{x})| > k\gamma] = \mathsf{ngl}(k)$. Then by a union bound, we have $\Pr[\exists i \in [m]\ s.t.\ |R_i(\vec{x})| > k\gamma] \leq m \cdot \mathsf{ngl}(k) = \mathsf{ngl}(k)$. Thus, we have

$$\Pr\left[|\vec{r} \cdot R(\vec{x})| > k \cdot m \cdot n^\lambda \cdot \gamma\right] < \Pr\left[|\vec{z} \cdot R(\vec{x})| > k \cdot m \cdot n^\lambda \cdot \gamma\right] = \Pr\left[\sum_{i\in[m]} n^\lambda \cdot |R_i(\vec{x})| > k \cdot m \cdot n^\lambda \cdot \gamma\right]$$

The event $\sum_{i\in[m]} n^\lambda \cdot |R_i(\vec{x})| > k \cdot m \cdot n^\lambda \cdot \gamma$ implies $\exists i \in [m]\ s.t.\ |R_i(\vec{x})| > k \cdot \gamma$ by an average argument. Thus we have

$$\Pr\left[\sum_{i\in[m]} n^\lambda |R_i(\vec{x})| > k \cdot m \cdot n^\lambda \cdot \gamma\right] \leq \Pr[\exists i \in [m]\ s.t.\ |R_i(\vec{x})| > k\gamma] = \mathsf{ngl}(k).$$

This completes the proof of the claim.

□

■

# 4 Hardness of MQ Problems Implies Pseudorandom Distributions

Recall that in the previous section, we claimed that the hardness of some family of MQ problems implies a pseudorandom distribution (Lemma 11). In this section, we are going to show that the hardness of more general families of MQ problems also implies a pseudorandom distribution. In particular, we obtain the following theorem.

**Theorem 22** *Let $k$ be the security parameter, $n, m, q$ be efficiently computable and polynomially bounded such that $q$ is an odd prime, $\chi$ is a distribution over $\mathbb{F}_q^{m\times n\times n}$, and $H \subseteq \mathbb{F}_q$.*

*Suppose for these parameters the problem $MQ(n, m, q, \chi, H)$ is hard to solve, then the following two distributions are computationally indistinguishable. $D_1 = (S, S(\vec{x}))$, $D_2 = (S, U_m)$, where $S \leftarrow MQ(n, m, q, \chi, H)$, $\vec{x} \leftarrow H^n$, and $U_m$ is a uniform distribution over $\mathbb{F}_q^m$.*

If we set $H$ to be $H_\beta$, and $\chi$ to be $\Phi_{\alpha,q}$ as the setting in Definition 9, then this version of the theorem directly becomes Lemma 11.

We prove the theorem by contradiction. For intuition, first we state our high level ideas and then delve into details. Suppose there exists a distinguisher $A$ that distinguishes $D_1$ and $D_2$, from here we want to construct an inverter $B$ that solves the MQ problem $(S, S(\vec{x}))$, which leads to a contradiction. We achieve this goal using the following strategy:

- First we show that from $A$, we can construct another algorithm $A'$ that distinguishes $D'_1 = (S, S(\vec{x}), \vec{r}, \langle \vec{r}, \vec{x}\rangle)$ and $D'_2 = (S, S(\vec{x}), \vec{r}, U)$ where $\vec{r} \in \mathbb{F}_q^n$ is a random vector, and $U$ is uniform over $\mathbb{F}_q$. For any $\vec{r} \in \mathbb{F}_q^n$, we can view $\langle \vec{r}, \vec{x}\rangle$ as the $\vec{r}$'s location of the (Hadamard) encoding of $\vec{x}$. The ability to distinguish $D'_1$ and $D'_2$ gives us a somewhat corrupted codeword of $\vec{x}$, i.e., the codeword is correct in at least a noticeable fraction of places over all $\vec{r}$'s.

- Then from $A'$, we construct an inverter $B$ that applies the list-decoding algorithm by the Goldreich-Levin Theorem to recover $\vec{x}$. We remind the reader that the Goldreich-Levin Theorem is essentially a decoding algorithm for the Hadamard code, which says (informally) that if given $f(\vec{x})$, for random $\vec{r}$'s one can distinguish $\langle \vec{r}, \vec{x}\rangle$ from a uniform element with noticeable probability, then one can invert $f$ with noticeable probability (for any function $f$).

However, when applying the Goldreich-Levin Theorem here, we encountered some subtleties. First the classical theorem [GL89] deals with the boolean field only (i.e. $q = 2$); thus it is not applicable in general cases. A generalized version of [GRS95] handles the case for large $q$'s, but it works only for the case where the input $\vec{x} \in \mathbb{F}_q^n$. It remains unclear for the case where $\vec{x}$ comes from a subset $H_\beta^n \subseteq \mathbb{F}_q^n$. Recently, Dodis *et al.* [DGK+10] proved a new version of the theorem that is essentially what we need in our setting. With it, we are able to implement the list-decoding algorithm in the second bullet above, and this completes the proof. We state the new version of Goldreich-Levin Theorem in the following, and then give a formal proof of Theorem 22.

**Theorem 23 (Goldreich-Levin [DGK$^+$10])** *Let $q$ be a prime, and let $H$ be an arbitrary subset of $\mathbb{F}_q$. Let $f : H^n \to \{0, 1\}^*$ be any (possibly randomized) function. If there exists a distinguisher $D$ that runs in time $t$ such that*

$$|\Pr[D(y, \vec{r}, \langle \vec{r}, \vec{s}\rangle) = 1] - \Pr[D(y, \vec{r}, u) = 1]| = \varepsilon,$$

*where $\vec{s} \leftarrow H^n, y \leftarrow f(\vec{s}), \vec{r} \leftarrow \mathbb{F}_q^n, u \leftarrow \mathbb{F}_q$,* *then there is an inverter $A$ that runs in time $t' = t \cdot \mathrm{poly}(n, |H|, 1/\varepsilon)$ (roughly $t' = t \cdot 128 \cdot n^2 \cdot |H|^2/\varepsilon^2$) such that*

$$\Pr[\vec{s} \leftarrow H^n, y \leftarrow f(\vec{s}) : A(y) = \vec{s}] \geq \frac{\varepsilon}{4q},$$

*if $q > 128|H|n/\varepsilon^2$. Otherwise, the success probability of $A$ is greater equal to $\frac{\varepsilon^3}{512 \cdot n \cdot q^2}$.*

**Proof.** (Theorem 22) We prove the theorem by contradiction. Suppose there exists a PPT adversary $A$ such that the following holds:

$$\left| \Pr_{D_1=(S,S(\vec{x}))}[A(S, S(\vec{x})) = 1] - \Pr_{D_2=(S,U_m)}[A(S, U_m) = 1] \right| > \varepsilon,$$

where $S \leftarrow MQ(n, m, q, \chi, H)$, $\vec{x} \leftarrow H^n$, $U_m$ is uniform over $\mathbb{F}_q^m$ and $\varepsilon = 1/\mathrm{poly}(k)$ is some noticeable quantity. From this we want to construct an inverter that solves the MQ problem $(S, S(\vec{x}))$ with some noticeable probability, and this leads to a contradiction. Here without of generality, we assume that the probability that $A$ outputs 1 on $D_1$ is higher, i.e. $\Pr_{D_1=(S,S(\vec{x}))}[A(S, S(\vec{x})) = 1] - \Pr_{D_2=(S,U_m)}[A(S, U_m) = 1] > \varepsilon$.

As mentioned in the intuition as above, we prove the theorem in two steps. In the first step we construct another distinguisher that on inputs $S, S(\vec{x})$ and a random $\vec{r}$, can tell $\langle \vec{r}, \vec{x} \rangle$ from random. More precisely we establish the following lemma:

**Lemma 24** *Let $A$ be a distinguisher to the distributions $D_1 = (S, S(\vec{x}))$ and $D_2 = (S, U_m)$, with advantage $\varepsilon \in [0, 1]$ and running in time $t \in \mathbb{N}$, where $(S, S(\vec{x})) \leftarrow MQ(n, m, q, \chi, H)$, and $U_m$ is uniform over $\mathbb{F}_q$. Then there exists a distinguisher to $A'$ running in time* $\mathrm{poly}(t, n, m, q)$ *that distinguishes $D_1' = (S, S(\vec{x}), \vec{r}, \langle \vec{r}, \vec{x} \rangle)$ and $D_2' = (S, S(\vec{x}), \vec{r}, U)$ within advantage* $\mathrm{poly}(\varepsilon, 1/q)$*, where $(S, S(\vec{x})) \leftarrow MQ(n, m, q, \chi, H)$, $\vec{r} \in \mathbb{F}_q^n$ and $U$ is uniform over $\mathbb{F}_q$ .*

**Proof.** (Lemma 24) Let $(S, S(\vec{x}), \vec{r}, z)$ be the input to $A'$ where $z$ comes from either $\langle \vec{r}, \vec{x} \rangle$ or $U$. Let $S = (R, L, \vec{b})$ be the coefficients of quadratic, linear and constant terms respectively. Then we define $A'$ as the following:

1. Sample $a_1, a_2, \ldots a_m \in \mathbb{F}_q$ uniformly, denote $[a_1, a_2, \ldots, a_m]^T$ as $\vec{a}$ (a column vector). Let $L' = \begin{bmatrix} - & a_1 \cdot \vec{r}^T & - \\ - & a_2 \cdot \vec{r}^T & - \\ & \vdots & \\ - & a_m \cdot \vec{r}^T & - \end{bmatrix}$ be an $m \times n$ matrix. Recall that $\vec{r}^T$ is a row vector of $n$ elements, and $a_i \cdot \vec{r}^T$ means every component of $\vec{r}^T$ is multiplied by $a_i$.

2. Set $S' = (R, L + L', \vec{b})$, and set $\vec{w} \in \mathbb{F}_q^m$ where $\vec{w}_i = a_i \cdot z$. Then $A'$ outputs $A(S', S(\vec{x}) + \vec{w})$.

To analyze the algorithm $A'$, we establish the following two claims:

**Claim 25** $\Pr_{D_1'}[A'(S, S(\vec{x}), \vec{r}, z) = 1] = \Pr_{D_1}[A(S, S(\vec{x})) = 1]$.

**Proof of claim:** For the case of the distribution $D_1'$, we observe that $\vec{w} = z \cdot \vec{a} = \langle \vec{r}, \vec{x} \rangle \cdot \vec{a}$. Thus $S(\vec{x}) + \vec{w} = S(\vec{x}) + \langle \vec{r}, \vec{x} \rangle \cdot \vec{a} = S(\vec{x}) + L' \cdot \vec{x} = S'(\vec{x})$. On the other hand, $S'$ is identically distributed with $S$ for the following observations. (1) for any fixed $L'$, a uniform $L$ is identically distributed with $L + L'$. (2) For $S = (R, L, \vec{b})$ and $S' = (R, L + L', \vec{b})$, the only distinction comes from the coefficients of linear terms, but the two are identically distributed.

Thus we have,

$$\Pr_{D_1'}[A'(S, S(\vec{x}), \vec{r}, z) = 1] = \Pr_{D_1'}[A(S', S'(\vec{x})) = 1] = \Pr_{D_1}[A(S, S(\vec{x})) = 1].$$

□

**Claim 26** $\Pr_{D_2'}[A'(S, S(\vec{x}), \vec{r}, z) = 1 | z = \langle \vec{r}, \vec{x} \rangle] = \Pr_{D_1'}[A(S, S(\vec{x}), \vec{r}, z) = 1]$.

**Proof of claim:** This follows since the conditional distribution of $D_2'|_{z=\langle \vec{r}, \vec{x} \rangle}$ is identical to $D_1'$. □

**Claim 27** $\Pr_{D_2'}[A'(S, S(\vec{x}), \vec{r}, z) = 1 | z \neq \langle \vec{r}, \vec{x} \rangle] = \Pr_{D_2}[A(S, U_m) = 1]$.

**Proof of claim:** Let $t \stackrel{\text{def}}{=} \langle \vec{r}, \vec{x} \rangle$. For the case of the distribution $D_2'$ conditioned on $z \neq t$, we observe that $\vec{w} = z \cdot \vec{a} = t \cdot \vec{a} + (z - t) \cdot \vec{a}$, where $z - t \neq 0$. Then we have $S(\vec{x}) + \vec{w} = S(\vec{x}) + t \cdot \vec{a} + (t - u) \cdot \vec{a} = S'(\vec{x}) + (t - u) \cdot \vec{a}$. For any given $S'(\vec{x})$, it is easy to see the distribution $(t - u) \cdot \vec{a}$ is uniformly random over $\mathbb{F}_q^m$ given $t - u \neq 0$ and $\vec{a}$ is uniform over $\mathbb{F}_q^m$. Thus $S(\vec{x}) + \vec{w}$ is identically distributed to $U_m$, and this completes the proof of the claim. □

Denote $P_{D_1'} \stackrel{\text{def}}{=} \Pr_{D_1'}[A'(S, S(\vec{x}), \vec{r}, z) = 1]$ and $P_{D_2'} \stackrel{\text{def}}{=} \Pr_{D_2'}[A'(S, S(\vec{x}), \vec{r}, z) = 1]$. Then we have

$$\begin{aligned}
& P_{D_1'} - P_{D_2'} \\
&= P_{D_1'} - (P_{D_2'}|_{(z=t)} \cdot \Pr[z = t] + P_{D_2'}|_{(z \neq t)} \cdot \Pr[z \neq t]) \\
&= P_{D_1'} - \left( P_{D_2'}|_{(z=t)} \cdot \frac{1}{q} + P_{D_2'}|_{(z \neq t)} \cdot \frac{q-1}{q} \right) \\
&= \frac{q-1}{q} \cdot P_{D_1'} - \frac{q-1}{q} \cdot P_{D_2'}|_{(z \neq t)} \\
&= \frac{q-1}{q} \left( \Pr_{D_1}[A(S, S(\vec{x})) = 1] - \Pr_{D_2}[A(S, U_m) = 1] \right) \\
&\geq \frac{q-1}{q} \cdot \varepsilon.
\end{aligned}$$

Clearly, the running time of $A'$ is $t + \text{poly}(n, m, q)$ and the advantage is greater equal to $\frac{q-1}{q} \cdot \varepsilon = \text{poly}(\varepsilon, 1/q)$.

■

The above proof of the lemma completes the first step. Then we proceed to the step 2. From the distinguisher $A'$, we would like to apply the Goldreich-Levin Theorem to construct an inverter to the instance $(S, S(\vec{x}))$. Now we need to set up the premises of Theorem 23. Let $f : H^n \to \{0,1\}^*$ be defined as the following: $f$ on input $\vec{x}$ samples $S \leftarrow MQ(n, m, q, \chi, H)$, and then outputs $(S, S(\vec{x}))$. By the above Lemma 24, we know $A'$ is a distinguisher that has advantage $\varepsilon' = \frac{q-1}{q} \cdot \varepsilon$

distinguishing $D'_1 = (S, S(\vec{x}), \vec{r}, \langle \vec{r}, \vec{x} \rangle) = (f(\vec{x}), \vec{r}, \langle \vec{r}, \vec{x} \rangle)$ and $D'_2 = (S, S(\vec{x}), \vec{r}, U) = (f(\vec{x}), \vec{r}, U)$. By Theorem 23, we know that there exists an inverter $B$ that inverts $f(\vec{x}) = (S, S(\vec{x}))$ with probability $\mathrm{poly}(\varepsilon', 1/n, 1/q)$. In our setting of parameters $n, m, q = \mathrm{poly}(k)$, and $\varepsilon = 1/\mathrm{poly}(k)$, and thus the inverter $B$'s success probability is $\mathrm{poly}(\varepsilon', 1/n, 1/q) = 1/\mathrm{poly}(k)$, and has a running time $\mathrm{poly}(k)$. This contradicts to the hardness assumption of MQ problem.

■

**Remark 28** We remark that Theorem 22 holds, under the premise that there exists some distribution $\chi$ and $H \subseteq \mathbb{F}_q$ such that the corresponding MQ problem is hard. In particular, if we set $q = 2$, $\chi$ to be uniform over $\mathbb{F}_q^{m \times n \times n}$ and $H = \mathbb{F}_q$, then this version of the assumption and Theorem is essentially the same as Theorem 2 plus 3 in [BGP06]. On the other hand, if we set $\chi$ to be some sparse distribution and $H = \mathbb{F}_q$, then the version of this theorem is essentially the same as Proposition 5 plus 6 in [LLY08]. Note that [BGP06, LLY08] present their theorems/propositions in concrete parameters, and in this paper, we present in an asymptotic way.

# 5 Key Encapsulation Mechanism

In the previous section, we constructed a public-key encryption for bits. However, this approach is not satisfactory when we want to encrypt a long message $M \in \{0,1\}^L$ for some large $L$. As discussed in the introduction, we can use a key encapsulation mechanism (KEM) to achieve better efficiency.

First, we recall how we can achieve this by the KEM technique: let Enc be any public-key encryption scheme for bits, and let $G : \{0,1\}^k \to \{0,1\}^{k+t}$ be a pseudorandom generator. To encrypt a long message $M \in \{0,1\}^L$, we first sample a seed $s \in \{0,1\}^k$ for the PRG, and then stretch the generator $G$[6] to get a pseudorandom string $G'(s) \in \{0,1\}^L$. Then we encrypt the seed by the public-key scheme and use the pseudorandom string as a one-time pad to XOR $M$. The resulting ciphertext becomes $(\mathrm{Enc}_{\mathsf{pk}}(s), G'(s) \oplus M)$.

In this paper, we observe that the MQ assumption implies a certain form of PRG. Thus, we can implement KEM under the same assumption as the one from which we construct the public-key encryption scheme. However, this type of PRG is somewhat non-standard, so we avoid using this term formally but discuss the issue in the following Remark.

In the next section, we are going to show how we can obtain the desired long pseudorandom string $G'(s)$, and then present the entire scheme in Section 5.2. Finally we sketch the proof of security, which follows from the folklore.

**Remark 29 (A discussion on the non-standard PRG)** Recall that in the standard definition of PRG, we say a function $G : \{0,1\}^n \to \{0,1\}^m$ is a PRG if (1) it is stretching i.e. $m > n$, (2) it is a deterministic and efficiently computable function, and (3) $G(\vec{x}) \approx_c U_m$.

However, it is unclear how we can obtain a single deterministic function $G$ from the implication of the MQ assumption that $(S, S(\vec{x})) \approx_c (S, U_m)$. The issue here is that $S$ is also a part in the distribution, so we cannot simply define $G = S(\cdot)$. On the other hand, if we consider a family of PRGs or a keyed-PRG where $S$ is taken into account as a key, then we can define $G_S(\vec{x}) = S(\vec{x})$. It is not hard to see that $G_S$ has some similar properties to standard PRGs: (1) it is stretching,

[6] We refer the readers to [Gol01] for the details of how to stretch a PRG.

(2) it is a deterministic and efficiently computable function given any key, and (3) even if the key is given, $(S, G_S(\vec{x}))$ is computationally indistinguishable from $(S, U_m)$.

In this viewpoint, obtaining a longer pseudorandom string can be obtained by stretching the keyed PRG given by the MQ assumption. The way we stretch the keyed PRG is essentially the same as that by which we stretch a standard PRG [BM84, Yao82]. The stream cipher QUAD [BGP06] was also constructed with the same spirit.

**Remark 30** We remark that KEM is a generic way to construct efficient public-key encryption schemes. As discussed in the introduction and the above, we know that a PRG plus any bit-encryption public encryption scheme is sufficient to achieve the task. In this paper, we observe that the MQ assumption implies an efficient constructions of PRGs and a public-key bit-encryption scheme, so we can obtain an efficient public-key encryption under one single assumption.

## 5.1 Longer Pseudorandom Strings

Recall that Lemma 11 states that $(S, S(\vec{x})) \approx_c (S, U_m)$. This means we can get a pseudorandom string $S(\vec{x}) \in \mathbb{F}_q^m$ by only sampling a shorter seed $\vec{x} \in H_\beta^n$. Note: $m > n$, and $H \subseteq \mathbb{F}_q$. To get a longer pseudorandom string, we can use the following iterative method (analogous to how we can stretch a PRG.)

**Definition 31** *Let $S \leftarrow MQ(n, m, q, \Phi_\alpha, H_\beta)$. For $\vec{x} \in H_\beta^n$, and let $(x_0, y_0) = S(\vec{x})$ where $\vec{x}_0 \in \mathbb{F}_q^n$, $\vec{y}_0 \in \mathbb{F}_q^{m-n}$ be the prefix $n$ elements and the suffix $m - n$ elements of $S(\vec{x})$ respectively.*

*Let $h : \mathbb{F}_q^n \to H_\beta^n$ be a hash function, and for $i \in \mathbb{N}$, we recursively define $(\vec{x}_i, \vec{y}_i) = S(h(\vec{x}_{i-1}))$ where $\vec{x}_i \in \mathbb{F}_q^n, \vec{y}_i \in \mathbb{F}_q^{m-n}$ (representing the prefix and suffix of $S(h(\vec{x}_{i-1}))$ respectively). Then we define $S_h^i(\vec{x}) = (\vec{y}_0, \vec{y}_1, \ldots, \vec{y}_i)$.*

$$\begin{tikzcd}
\text{Initial input} = \vec{x} \arrow[rr, "S(\vec{x})"] & {} \arrow[d] & x_0 \arrow[rr, "S(h(\vec{x}_0))"] & {} \arrow[d] & x_1 \arrow[rr, "S(h(\vec{x}_1))"] & {} \arrow[d] & x_2 & \cdots \\
\text{output:} & y_0 & & y_1 & & y_2 & & \cdots
\end{tikzcd}$$

Then we are going to argue that for any $i \leq \text{poly}(k)$, we have $(S, S_h^i(\vec{x})) \approx_c (S, U_{(m-n)\cdot(i+1)})$, given $(S, S(\vec{x})) \approx_c (S, U_m)$. This means, we can get an arbitrarily long (polynomially bounded) pseudorandom string $S_h^i(\vec{x})$ from an initial random seed $\vec{x}$.

First we give an intuition why this works and why we need a hash function. Let us take $i = 2$ for simplicity of exposition. From $(S, S(\vec{x})) = (S, \vec{x}_0, \vec{y}_0) \approx_c (S, U_m) = (S, U_n, U_{m-n})$, we know that by applying $h$ on one part of the distributions we still have $(S, h(\vec{x}_0), \vec{y}_0) \approx_c (S, h(U_n), U_{m-n}) = (S, \vec{x}, U_{m-n})$ since $h(U_n)$ is distributed the same as $\vec{x} \in H_\beta^n$. Then we have $(S, S(h(\vec{x}_0)), \vec{y}_0) \approx_c (S, S(\vec{x}), U_{m-n}) \approx_c (S, U_m, U_{m-n})$. This completes the intuition.

We remark that for the known construction (of how to stretch PRGs to obtain more pseudorandom bits)[BM84, Yao82], we do not need a hash function, or in other words we let $h$ to be the identity mapping from $\mathbb{F}_q \to \mathbb{F}_q$. In our setting, we need $h$ since the distribution $(S, S(\vec{x}))$ is only guaranteed to be pseudorandom if $\vec{x}$ is random over $H_\beta^n$. Here we only need $h$ to reinterpret each $\vec{x}_i \in \mathbb{F}_q^n$ as an element in $H_\beta^n$. Thus the hash function is easy to implement as the following remark.

**Remark 32** The only property we need for $h$ is $h(U_n)$ outputs a uniformly random element in $H_\beta^n$. We can view elements in $\mathbb{F}_q^n$ as strings in $\{0,1\}^{\lceil n\cdot\log q\rceil}$, and those in $H_\beta^n$ as strings in $\{0,1\}^{\lceil n\cdot\log(2\beta+1)\rceil}$. Then $h$ on input $\vec{z}\in\mathbb{F}_q^n$ first interprets $\vec{z}$ as an $\lceil n\cdot\log q\rceil$-bit string, and then outputs the first $\lceil n\cdot\log(2\beta+1)\rceil$ bits interpreted as an element in $H_\beta^n$. It is not hard to see this construction meets our requirement.

Then we are able to achieve the following theorem.

**Theorem 33** *Let $k$ be the security parameter. Assuming the MQ problem $MQ(n,m,q,\Phi_\alpha,H_\beta)$ is hard, and let $h:\mathbb{F}_q^n\to H_\beta^n$ be a (randomized) hash function such that $h(\vec{z})$ maps a uniformly random $\vec{z}\in\mathbb{F}_q^n$ to a uniformly random $\vec{y}\in H_\beta^n$.*

*Then for any $i=\mathrm{poly}(k)$, $(S,S_h^i(\vec{x}))$ is computationally indistinguishable to $(S,U_{(m-n)\cdot(i+1)})$, where $S\leftarrow MQ(n,m,q,\Phi_\alpha,H_\beta)$, $\vec{x}\leftarrow H_\beta^n$, and $U_{(m-n)\cdot(i+1)}$ is uniform over $\mathbb{F}_q^{(m-n)\cdot(i+1)}$.*

**Proof.** We prove the theorem by contradiction. Suppose there exists $i=\mathrm{poly}(k)$ such that one can distinguish $(S,S_h^i(\vec{x}))$ and $(S,U_{(m-n)\cdot(i+1)})$, then one can distinguish $(S,S(\vec{x}))$ and $(S,U_m)$, which leads to a contradiction to Lemma 11.

More formally, let $A$ be a PPT distinguisher that has advantage $\varepsilon=1/\mathrm{poly}(k)$ distinguishing $(S,S_h^i(\vec{x}))$ and $(S,U_{(m-n)\cdot(i+1)})$, then we are going to design a distinguisher $B$ for $(S,S(\vec{x}))$ and $(S,U_m)$ by a hybrid argument.

Given a system $S$ and a hash function $h$ as stated, we define hybrid distributions for $j\in[i]$ as follows:

$$H_j=\left(S,U_{m-n}^1,\ldots,U_{m-n}^j,S_h^{i-j}(h(U_n))\right),$$

and we define

$$H_0=\left(S,S_h^i(h(U_n))\right),H_{i+1}=\left(S,U_{m-n}^1,\ldots,U_{m-n}^{i+1}\right),$$

where $U_n$ is uniformly random over $\mathbb{F}_q^n$, and $U_{m-n}^\ell$'s are independent copies of random vectors over $\mathbb{F}_q^{m-n}$ for $\ell\in\mathbb{N}$.

First we observe that $H_{i+1}=(S,U_{(m-n)\cdot(i+1)})$ by its definition. On the other hand, since $h$ maps a uniform random vector to a uniform random vector in $H_\beta^n$, and thus $h(U_n)$ is identically distributed with $\vec{x}$. This implies $H_0=(S,S_h^i(\vec{x}))$.

From the advantage of $A$ between $H_0$ and $H_{i+1}$, by an average argument there exists an $i^*\in[i]$ such that $\Pr[A(H_{i^*})=1]-\Pr[A(H_{i*+1})=1]>\varepsilon/(i+1)$. We remark that here we assume that the distinguisher $B$ has the non-uniformity advice of $i^*$ for simplicity of presentation, but we can remove it by a random guess from $[i]$ as the standard hybrid argument proof.

Let $(S,\vec{z})$ be the input of $B$ where $\vec{z}$ comes from either $S(\vec{x})$ or $U_m$. Denote $\vec{z}=(\vec{z}_1,\vec{z}_2)$ where $\vec{z}_1$ is the prefix of the $n$ elements and $\vec{z}_2$ is the suffix of the rest. Then $B$ samples $u_1,u_2,\ldots u_{i^*}\leftarrow U_{n-m}$ independently, and then $B$ outputs $A\left(S,u_1,\ldots u_{i^*},\vec{z}_2,S_h^{i-i^*-1}(h(\vec{z}_1))\right)$.

In the following claims, we are going to argue that the two different input distributions of $B$ can be translated to $H_{i^*}$ and $H_{i^*+1}$ respectively, and thus $A$'s advantage can be carried to $B$.

**Claim 34** *If $\vec{z}$ comes from the distribution $U_m$, then we have*

$$\left(S,u_1,\ldots u_{i^*},\vec{z}_2,S_h^{i-i^*-1}(h(\vec{z}_1))\right)=H_{i^*+1}.$$

**Proof of claim:** This follows directly from that the distribution of $\vec{z}_1$ is $U_n$. □

**Claim 35** *If $\vec{z}$ comes from the distribution $S(\vec{x}) = (\vec{z}_1, \vec{z}_2)$, then we have*

$$\left(S, u_1, \ldots u_{i^*}, \vec{z}_2, S_h^{i-i^*-1}(h(\vec{z}_1))\right) = H_{i^*}.$$

**Proof of claim:** We observe that in this case, $(\vec{z}_1, \vec{z}_2) = (\vec{x}_0, \vec{y}_0)$. Then by the definition, we have $S^{j+1}(\vec{x}) = (y_0, S^j(h(\vec{x}_0))$ for any $j \in \mathbb{N}$. Thus we have

$$LHS = \left(S, u_1, \ldots u_{i^*}, S_h^{i-i^*}(\vec{x})\right) = \left(S, u_1, \ldots u_{i^*}, S_h^{i-i^*}(h(U_n))\right) = H_{i^*}.$$

Note that $h(U_n)$ is identical to $\vec{x}$, and thus the above equation holds.

□

Thus, the advantage of $A$ carries to $B$, and $B$ can distinguish the two distributions with advantage $\varepsilon/(i+1) = 1/\text{poly}(k)$, which leads to a contradiction. ■

## 5.2 Construction of the KEM Scheme

In previous sections, we have constructed the bit encryption scheme $\mathcal{E} = (\text{KeyGen}(\cdot), \text{Enc}(\cdot), \text{Dec}(\cdot))$ described in section 3.2, and the pseudorandom generator above. Here we describe a KEM scheme $\mathcal{E}_{\mathsf{KEM}} = (\text{KeyGen}_{\mathsf{KEM}}(\cdot), \text{Enc}_{\mathsf{KEM}}(\cdot), \text{Dec}_{\mathsf{KEM}}(\cdot))$ that can encrypt messages with un-prespecified lengths (polynomially bounded).

- $\text{KeyGen}_{\mathsf{KEM}}(1^k)$: run $\text{KeyGen}(1^k)$. In particular, the algorithm chooses public parameters $n, m, q, \Phi_\alpha, H_\beta$ in the range as stated in the MQ assumption, and also a hash function $h : \mathbb{F}_q^n \to H_\beta^n$ with the property $h(U_n)$ being uniform over $H_\beta^n$ as discussed in the above section. Then it samples a random instance $(S, S(\vec{x})) \leftarrow MQ(n, m, q, \Phi_\alpha, H_\beta)$, and deontes $\vec{y} = S(\vec{x})$. Then it sets $\mathsf{pk} = (S, \vec{y})$, $\mathsf{sk} = \vec{x}$.
- For any $L = \text{poly}(k)$, and any message $M \in \mathbb{F}_q^L$, $\text{Enc}_{\mathsf{KEM}}(M)$ does the following: the algorithm samples $\vec{s} \in H_\beta^n$, and computes $c_i = \text{Enc}(\mathsf{pk}, \vec{s}_i)$ for $i \in [n]$. Then let $t = \lceil L/(m-n) \rceil$, and compute $c^* = M \oplus S_h^t(\vec{s})$.[7] The resulting ciphertext will be $c = (c_1, c_2, \ldots, c_n, c^*)$.
- $\text{Dec}_{\mathsf{KEM}}(c)$: the algorithm computes $\vec{s}$ by running $\text{Dec}(\mathsf{sk}, c_i)$ for $i \in [n]$. Then it outputs $M = c^* \oplus S_h^t(\vec{s})$.

## 5.3 The Analysis

Here we briefly argue the correctness and the security of the scheme $\mathcal{E}_{\mathsf{KEM}}$. The correctness follows directly from that of the bit encryption scheme $\mathcal{E}$. The security follows from the folklore. Here we informally outline the proof.

- By the semantic security of $\mathcal{E}$, we know that $(\mathsf{pk}, \text{Enc}(\mathsf{pk}, \vec{s}), \vec{s}) \approx_c (\mathsf{pk}, \text{Enc}(\mathsf{pk}, \vec{s}), \vec{r})$ where $\vec{s}, \vec{r}$ are i.i.d and uniformly random from $H_\beta^n$.

[7] Here $\oplus$ means we add two vectors component-wise. That is, let $\vec{a}, \vec{b} \in \mathbb{F}_q^L$, then we say $\vec{a} \oplus \vec{b} = [\vec{a}_1 + \vec{b}_1, \vec{a}_2 + \vec{b}_2, \cdots, \vec{a}_L + \vec{b}_L]^T$.

- Then we know that $(\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), S_h^t(\vec{s})) \approx_c (\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), S_h^t(\vec{r}))$ since applying a function on a part of the distributions won't affect computational indistinguishability.

- Then we have $(\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), S_h^t(\vec{r})) \approx_c (\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), U_L)$ by the property of $S_h^t(\vec{r})$. Note that $\vec{r}$ now is an independent sample from $\vec{s}$.

- Thus, we have any message $M$, $(\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), S_h^t(\vec{s}) \oplus M) \approx_c (\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), U_L \oplus M) \approx (\mathsf{pk}, \mathrm{Enc}(\mathsf{pk}, \vec{s}), U_L)$. Thus, we have the fact that for any two messages $M_1, M_2$, we have $(\mathsf{pk}, \mathrm{Enc}_{\mathsf{KEM}}(M_1)) \approx_c (\mathsf{pk}, \mathrm{Enc}_{\mathsf{KEM}}(M_2))$.

Putting everything together, we obtain the following theorem.

**Theorem 36** *The scheme above $\mathcal{E}_{\mathsf{KEM}}$ is a semantically secure encryption scheme.*

### 5.4 Concrete Parameters

Our goal here is to instantiate Theorem 36 with concrete parameters. Here, we exhibit two sets of parameters (for proven security levels $2^{80}$ and $2^{128}$) based on a conservative estimate of the hardness of MQ systems (i.e., assuming the general applicability of sparse matrix solvers in XL [YCBC07]), and no particular effort in optimization.

Our security level aims for time $2^{80}$, and $\varepsilon = 2^{-10}$, i.e., no adversary within running time $2^{80}$ can distinguish two ciphertexts with advantage better than $2^{-10}$. Since our construction uses the KEM mechanism, we need parameters for (1) $(S, S_h^t(\vec{x}))$ to be a PRG some length $L$, and (2) $\mathcal{E}$ to be a semantically secure bit-encryption scheme. It follows from a standard argument that the KEM security achieves this level (with a slight loss) once both the underlying PRG and the encryption scheme achieve this level of security.

First we analyze the security of the PRG $(S, S_h^t(\vec{x}))$ by calculating the concrete parameters of Theorem 33. Suppose there exists an adversary $A$ running in time $T$ with advantage $\varepsilon$ that can distinguish it from random, then by the analysis of Theorem 33, there is an adversary $B$ running in time $T' = T + m \cdot n^2 \cdot \log^2 q$ with advantage $\varepsilon' = \varepsilon/t$ (recall that $t = \lceil L/(m-n) \rceil$) that distinguishes $(S, S(\vec{x}))$ from $(S, U_m)$. Then by Theorem 22, we know that there exists an adversary $C$ running in time $T'' = 128 \cdot |H_\beta|^2 \cdot (n^2/\varepsilon'^2) \cdot T'$ with probability $\varepsilon'' = \varepsilon'/4q$ that solves the MQ hard problem. (Our choice of $q$ is large enough for Theorem 23 such that we can apply this bound for $\varepsilon''$.)

Then we analyze the security of the encryption scheme. The encryption scheme needs to encrypt $n \cdot \log q$ bits. The security follows from the indistinguishability of $(S, S(\vec{x})) \approx_c (S, U_m)$. That is, suppose there is an adversary running in time $T$ with advantage $\varepsilon$ that breaks the $n \cdot \log q$-bit encryption scheme, by a union bound argument, the adversary can break the one-bit version in time (almost) $T$ with advantage $\varepsilon' = \varepsilon/n \cdot \log q$. Then by Theorem 22, there exists an adversary $B$ running in time $T' = 128 \cdot |H_\beta|^2 \cdot n^2/\varepsilon'^2 \cdot T$ with probability $\varepsilon'/4q$ that solves the MQ hard problem.

Now we are ready to set the parameters for the (bit) encryption scheme. Recall we need the conditions:

1. $k \cdot \alpha \cdot n^{(2+\lambda)} \cdot m \cdot \beta^2 \le q/4$.

2. $m \cdot \log(2n^\lambda + 1) \ge (n+1) \cdot \log q + 2k$.

3. $n, m, q, \alpha, \beta$ satisfy the condition in the MQ assumption such that $MQ(n, m, q, \Phi_\alpha, H_\beta)$ is hard to solve.

Here we consider two sets of parameters of the MQ system:

| Case | $k$ | $n$ | $m$ | $\alpha$ | $\beta$ | $q$ | Hardness $(T,\, \nu)$ |
|---|---|---|---|---|---|---|---|
| 1 | 12 | 200 | 400 | 10 | 2 | $180313175469726327885 19 \approx 2^{74}$ | $2^{156}$, $2^{-100}$ |
| 2 | 12 | 256 | 512 | 10 | 2 | $523244027957626787248 73 \approx 2^{76}$ | $2^{205}$, $2^{-104}$ |

It is not hard to verify that the first two conditions of the encryption scheme are met. We estimate the hardness using XL with a sparse matrix solver, with the complexity evaluated as in [YCBC07]. We forgo more advanced Gröbner bases method such as $\mathbf{F_5}$ because they have the same asymptotic behavior but much larger memory footprint, which we believe to be critical for large enough problem sizes. Here a hardness of $T, \nu$ means: no solver running in time less than $T$ can solve the system with probability better than $\nu$.

Then we are ready to analyze the security of the bit-encryption scheme.

- We claim that the security of the first case is $2^{82}$, $2^{-11}$. That is no adversary running in time $2^{82}$ can distinguish two ciphertexts with advantage better than $2^{-11}$. Suppose not, then there exists a solver that solves the MQ system running in time $(128 \cdot |H_\beta|^2 \cdot n^2/\varepsilon'^2) \cdot 2^{82} \approx 2^{156}$ (where $\varepsilon' = \varepsilon/n \log q$) with probability better than $2^{-11}/4q \approx 2^{-87}$. This contradicts the hardness estimation.

- We claim that the security of the second case is $2^{130}$, $2^{-11}$. That is no adversary running in time $2^{130}$ can distinguish two ciphertexts with advantage better than $2^{-11}$. Suppose not, then there exists a solver that solves the MQ system running in time $(128 \cdot |H_\beta|^2 \cdot n^2/\varepsilon'^2) \cdot 2^{130} \approx 2^{205}$ (where $\varepsilon' = \varepsilon/n \log q$) with probability better than $2^{-11}/4q \approx 2^{-89}$. This contradicts the hardness estimation.

Now we are ready to hardness of the PRG for the two cases. We consider the block length $L = 2^{20}$ (1 Mb).

- We claim that the security of the first case is $2^{85}$, $2^{-11}$. That is no adversary running in time $2^{85}$ can distinguish the output of the PRG from the uniform distribution with advantage better than $2^{-11}$. Suppose not, then there exists a solver that solves the MQ system running in time $(128 \cdot |H_\beta|^2 \cdot n^2/\varepsilon'^2) \cdot 2^{85} \approx 2^{156}$ (where $\varepsilon' = \varepsilon/t$, $t = L/(m-n)$) with probability better than $2^{-11}/4tq \approx 2^{-100}$. This contradicts the hardness estimation.

- We claim that the security of the second case is $2^{134}$, $2^{-11}$. That is no adversary running in time $2^{134}$ can distinguish the output of the PRG from the uniform distribution with advantage better than $2^{-11}$. Suppose not, then there exists a solver that solves the MQ system running in time $(128 \cdot |H_\beta|^2 \cdot n^2/\varepsilon'^2) \cdot 2^{134} \approx 2^{205}$ (where $\varepsilon' = \varepsilon/t$, $t = L/(m-n)$) with probability better than $2^{-11}/4tq \approx 2^{-101}$. This contradicts the hardness estimation.

Thus, by a union bound argument, we are able to achieve the following table:

| Case | Hardness of MQ | Security of Enc | Security of PRG | Security of KEM |
|---|---|---|---|---|
| 1 | $2^{156}$, $2^{-100}$ | $2^{87}$, $2^{-11}$ | $2^{85}$, $2^{-11}$ | $2^{85}$, $2^{-10}$ |
| 2 | $2^{205}$, $2^{-104}$ | $2^{130}$, $2^{-11}$ | $2^{134}$, $2^{-11}$ | $2^{130}$, $2^{-10}$ |

We remark the tuple $(T, \varepsilon)$ in each cell means for any adversary running in time $T$ has advantage (or success probability) less than $\varepsilon$.

# 6 Empirical Studies

In this section, we present our empirical results to confirm the hardness of the assumption in Definition 9, which we restate in the following.

**Definition 37 (MQ Hardness Assumption (restated))** *Let $k$ be the security parameter. For every constant $c > 1 \in \mathbb{N}$, every efficiently computable and polynomially bounded $n, m, q : \mathbb{N} \to \mathbb{N}$, $\alpha : \mathbb{N} \to [-q/2, q/2]$ and every $0 < \beta \leq [q/2]$ such that (1) $m = cn$, (2) $q$ is prime, (3) $\alpha = O(1)$, let $\Phi_\alpha$ be the distribution of $m \times n \times n$ identical independent discrete Gaussian distribution $D_\alpha$'s with mean $0$, standard deviation $\alpha$, namely, each $D_\alpha$ samples $z \leftarrow N(0, \alpha^2)$ (normal distribution with mean $0$, and standard deviation $\alpha$), and then outputs $\lfloor z \rceil \pmod q$, and let $H_\beta = \{-\beta, -\beta + 1 \ldots, \beta - 1, \beta\}$.*

*Then the problem $MQ(n, m, q, \Phi_\alpha, H_\beta)$ is hard to solve.*

We know that $MQ(n, m, q, U, \mathbb{F}_q)$ is basically the standard MQ problem which most people believe to be hard (cf. [BGP06]). We make the following observations about the hardness of the problem as we make $\Phi_\alpha$ a discrete Gaussian and restrict the input variables to a small $H_\beta$.

**The effect of restricting the input** We consider $H_1$ and $H_2$ specifically because we used $\beta = 2$ as our choice of concrete parameters. Also, the problem for $\beta = 1$ is believed to be the easiest one, so if the easiest one is hard, then all the other problems with larger $\beta$'s are hard as well.

Fixing all inputs to $H_1$ or $H_2$, which makes $x_j \in \{\pm 1, 0\}$ (or resp. $x_j \in \{\pm 2, \pm 1, 0\}$) is exactly the same as adding equations $x_j^3 = x_j$ (resp. $x_j(x_j^2 - 1)(x_j^2 - 4) = 0$) for every $j$ to the system. When we solve a system via Gröbner bases methods (XL or $\mathbf{F_4/F_5}$), these "extra" equations eliminate all monomials with any of the exponents 3 (resp. 5) or above. This also happens when we solve a random system in a small field such as GF(3) (or GF(5)), where the Frobenius equations $x_j^3 = x_j$ (resp. $x_j^5 = x_j$) [8] similarly cuts down the number of monomials.

Intuitively, since Gröbner bases methods involve linearization and elimination of monomials, fewer monomials means less difficult to solve. [This is in fact why solving a random system of a given size is easier for smaller fields.]

We observe that solving a quadratic system with the variables restricted to $H_1$ (or resp. $H_2$), is similar to solving a system of the same size over a smaller field such as $\mathbb{F}_3$ (resp. $\mathbb{F}_5$). They take almost the same number of arithmetic operations (multiplications). We remark that the total complexity depends on the complexity of the multiplication, which is about poly$(\log q)$, where $q$ is the field size. For small fields, the difference is not significant.

In our experiments, we test the hardness of solving random systems over $\mathbb{F}_3, \mathbb{F}_5$. We observe that solving systems in these small fields is also hard, and thus we can conclude that solving systems in large field with restricted inputs is also hard.

**The effect of the Gaussian Coefficients** We also ran experiments where the quadratic coefficients sampled as discrete normals for a wide spread of standard deviations $\alpha$. We noted that for $\alpha$ as little as only 1/4 (i.e., only about 4.5% of the coefficients are non-zero, and most of the remainder are $\pm 1$), and fixing the input to $\{\pm 1, 0\}$, solving the system via a Gröbner bases method is still as difficult or more so than solving the same-sized system over $\mathbb{F}_3$.

[8] This equation is equivalent to $x_j(x_j^2 - 1)(x_j^2 - 4) = 0$ in the field $\mathbb{F}_5$

The above is expected given that [LLY08] already noted empirically that even if we sample the quadratic coefficients of the instance $S$ in a biased fashion, for example defined to be sparse, solving the problem is still almost equally hard as long as the system does not “get too sparse”.

**Alternatives to Gröbner Bases** We should note that while there are some methods such as SAT-solvers which does well in solving certain types of equations, these all have one or more of the following drawbacks:

- Some are restricted to GF(2) or perhaps by extension small binary fields.
- Some are restricted to heavily cliqued systems, where each equation does not contain most of the variables.
- Some are restricted to very sparse systems where the number of terms are linear or sub-linear to the number of variables.
- Some are restricted to very overdetermined systems.

Therefore, it should not be surprising that we do not find anything in the literature which seems to offer any particular advantage against our chosen MQ systems.

**Operating Degree in Gröbner Bases Methods and Hardness** Given the best way of solving our special MQ problems still seems to be Gröbner bases, we note that the difficulty to solve a system of equations can often gauged by one single standard metric, the operating degree $D$.

The running time of a modern Gröbner bases method at degree $D$ is bounded by $(T(D,n))^\omega$ where $\omega$ is the order of matrix multiplication and $T$ is the number of monomials which is equal to $\binom{n+D}{D}$ for large fields and $\binom{n}{D}$ for $q = 2$. Typically, $D_{reg}$ is asymptotically linear in $n$ [BFS04, BFSY05, YCC04], which means that $T(D,n)$ will be singly exponential in $n$. This is consistent with the impression that solving a system of equations is hard.

In the following we describe the experiments we ran to validate our observations above and our contention that $MQ(n, m, q, \Phi_\alpha, H_\beta)$ is in fact hard.

**Effects of Guessing and Probabilistic Solution** A typical way of solving MQ-type problems is to fix (guess at) some of the variables and then solve the rest via XL or $\mathbf{F_4/F_5}$.

We note here that for most parameters within the range discussed in this paper, such Fixing before solving does not result in a decrease in total workload. This point was mentioned before (e.g., in [YCC04]) that for any $q$ and $\omega$, one should guess at variables until the ratio $m/n$ asymptotically reaches an optimal point which depends only on $q$ and $\omega$, after no further guessing can decrease the total amount of work done. This limit is typically already exceeded for $\omega = 2 + o(1)$ (i.e., XL with Wiedemann) and $m/n \geq 2$.

The above lemma relies implicitly on the fact that if it should not be possible to demonstrate lack of solutions of a system in a substantially shorter time than it is to solve a similar system with a solution. We also verify this empirically by running XL and $\mathbf{F_4/F_5}$ on systems similar to those we are interested, i.e., there would be $n$ variables, $m = cn$ equations where $c$ is between 1.5 and 3, the equations have their quadratic coefficients in discrete normal distributions, and add the equation $x_j^3 = x_j$ or $x_j(x_j^2 - 1)(x_j^2 - 4) = 0$ for every $i$. The only difference is that these extra test systems do not have solutions. We verify that MAGMA's $\mathbf{F_4}$ does take longer to terminate when the system is self-contradictory than when it has solution.

Given all the above, if we assume that XL with a sparse solver is essentially the best way to solve MQ, then the minimum time $T$ in which we can solve a system with probability $\nu = q^{-k}$, can be evaluated by the complexity of XL after we guess at $k$ variables. This explains how we computed the "hardness $(T, \nu)$ in Sec. 5.4.

**Description of Experimentation** To evaluate the hardness of the problem, we use the best-known and commercially available system-solver to our knowledge, which is the $F_4$ implementation in `MAGMA-2.17-7`, as produced by University of Sydney. The experiments were run on AMD Opteron servers at 2.1GHz, and Intel Xeon CPU E7-7550 at 2.0GHz with 256 GB of RAM each. Each instance was sampled at least 3 times and averaged.

Let $U$ denote the uniform distribution over quadratic coefficients. Since $MQ(n, m, q, U, \mathbb{F}_q^n)$ is more well-studied in the literature since [BGP06], and is broadly believed as a hard problem, it is a good reference with which the new assumption compares. We ran the following tests over many different instances:

1. For simplicity, we set $n = k$, $m/n = \text{const}$. The ratio is set to 2 and 2.5, then we run the solver on $MQ(n, m, q, \Phi_\alpha, H_\beta)$ for many different $(n, q, \alpha, \beta)$'s.

2. We also run for comparison running times and memory use the following:

   - $MQ(n, m, q, U, \mathbb{F}_q)$, where $U$ denotes the uniform distribution over the quadratic terms, and the input $\vec{x}$ comes from $\mathbb{F}_q^n$.
   - $MQ(n, m, q, U', \mathbb{F}_q)$, where $U'$ denotes $m$ random quadratics with coefficients in Gaussian distribution $\Phi_\alpha$, *plus $n$ random equations of degree $2\beta + 1$ (simulating the restriction to $H_\beta$),* and the input $\vec{x}$ comes from $\mathbb{F}_q^n$.
   - $MQ(n, m, 2\beta + 1, U, \mathbb{F}_{2\beta+1})$, where we simply restrict the variables to a much smaller field with $2\beta + 1$ elements (which is legal for $\beta = 1, 2, 3$ since $2\beta + 1$'s are primes for these $\beta$'s ).

From our experiment results, we observe the following:

- For fixed $\beta = 1$, a fixed $q$, $m = \Theta(n)$, the time and space complexity of solving the system $MQ(n, m, q, U'(\Phi_\alpha), H_\beta)$ is larger than those of $MQ(n, m, 2\beta + 1, U, \mathbb{F}_{2\beta+1})$, which grow exponentially with $n$. It is always smaller than those of $MQ(n, m, q, U, \mathbb{F}_q)$, which also grow exponentially. These are all in line with expectations.

- As expected, $\beta = 1$ is by far the simplest case, since it allows the substitutions $x_i^3 = x_i$ and everything becomes much easier. In fact as $\beta$ increases above 1, the running time quickly gets very close to that of $MQ(n, m, q, U, \mathbb{F}_q)$ (a system with no restrictions on $\vec{x}$).

- The complexity is almost indifferent to $\alpha$. [LLY08] mentioned a similar phenomenon: As long as there is enough variation such that enough of the coefficients are non-zero, the complexity seems to be exponential in $n$ for fixed $m/n$.

- For different $q$'s, the complexities grow when $q$ becomes significantly larger. This is not surprising since the arithmetic operations take more time when $q$ grows, and in particular the dependency is polynomial in $\log q$. In fact, the algorithm that MAGMA solves equations is the same for any input finite field, so we can expect the numbers of multiplications are roughly the same. Thus, the complexity for different $q$'s mainly depends on the complexity of its fundamental operations. Our test cases for different $q$'s confirm this observation.

In our assumption, we assume the hardness up to some super-polynomial (i.e. against $k^{\omega(1)}$ adversaries with advantages up to $\mathsf{ngl}(k)$). The above observations suggest that the hardness of $MQ(n, m, q, U', H_\beta)$ depends on $n$ mostly and in an exponential way. In our parameterization, we let $n = \mathrm{poly}(k)$, and thus the hardness also depends on $k$ in an exponential way. This give a direct justification of the hardness of the assumption, and moreover, the MQ problems may be even harder than we've assumed; in the first item, we also compares our assumption with a believed hard problem, and shows that the problem in our assumption is not easier. This gives another confirmation.

## 6.1 Data

In this section, we present some of our data. All the time units are second(s), and memory units are Megabytes (Mb). We attach several sample of our data here with $\beta = 1, 2$ and $m/n = 2, 2.5$, and remark that all the others have similar natures. Then we also present comparison tables between different cases. In particular, the tables here show that the complexities of all cases are exponential. In addition, Table 4 (reps. Table 5) compares all instances to $MQ(n, m, 3, U, \mathbb{F}_3)$ (reps. $MQ(n, m, 5, U, \mathbb{F}_5)$) in $m = 2n$, which is the easiest one in our intuition but still has an exponential behavior. Table 9 (reps. Table 10) compares all instances to $MQ(n, m, 3, U, \mathbb{F}_3)$ (reps. $MQ(n, m, 5, U, \mathbb{F}_5)$) also shows the same fact in the case $m = 2.5n$. This gives evidence of the hardness for other cases. At the last part of the experimental result, according to Table 11, it takes no advantages, or even more time (Table 12), to calculate a system without solutions. This means that to guess some value of variables randomly also costs much in exchange.

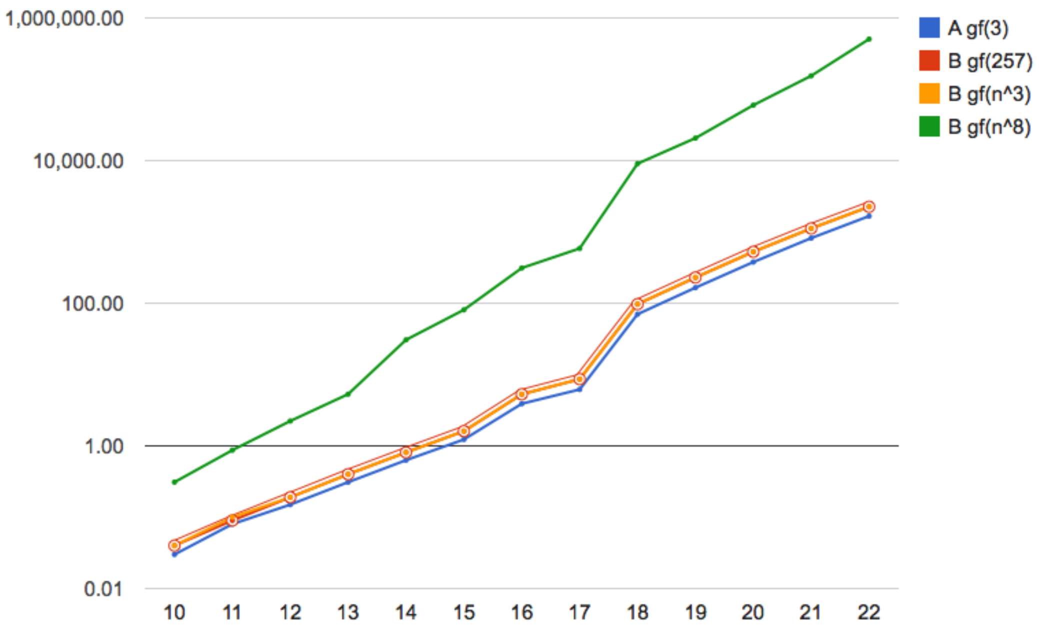

Figure 1: Trends of time of Case A, where $m = 2n$ with $\mathbb{F}_3$ and Case B, with variables in $\{-1, 0, 1\}$.

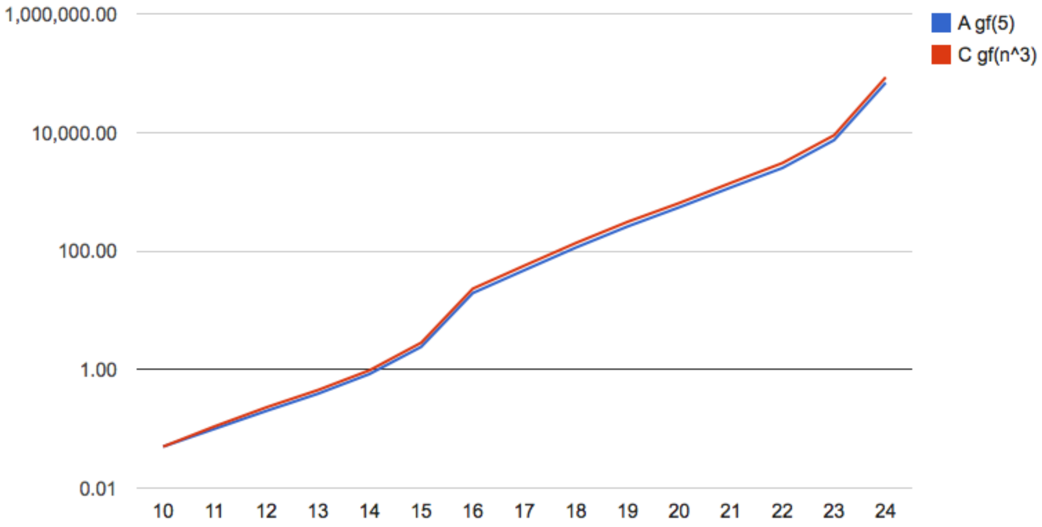

Figure 2: Trends of time of Case A where $m = 2n$ with $\mathbb{F}_5$ and Case C with variables in $\{-2, -1, 0, 1, 2\}$.

| $q$ | $\mathbb{F}_3$ | | | $\mathbb{F}_5$ | | | $\mathbb{F}_{257}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.03 | 9.69 | 6 | 0.05 | 10.75 | 4 | 0.04 | 10.72 |
| 11 | 4 | 0.08 | 10.78 | 6 | 0.10 | 10.78 | 4 | 0.09 | 10.78 |
| 12 | 4 | 0.15 | 10.84 | 6 | 0.20 | 11.09 | 4 | 0.20 | 10.84 |
| 13 | 4 | 0.31 | 12.02 | 6 | 0.39 | 12.30 | 4 | 0.42 | 11.97 |
| 14 | 4 | 0.63 | 13.07 | 6 | 0.84 | 13.94 | 4 | 0.92 | 14.23 |
| 15 | 4 | 1.23 | 14.75 | 6 | 2.41 | 16.02 | 4 | 2.76 | 16.12 |
| 16 | 4 | 3.90 | 17.19 | 6 | 19.47 | 48.59 | 5 | 22.06 | 42.41 |
| 17 | 4 | 6.18 | 19.73 | 6 | 47.38 | 74.30 | 5 | 55.36 | 68.91 |
| 18 | 5 | 70.19 | 70.87 | 6 | 114.30 | 115.94 | 5 | 134.08 | 114.84 |
| 19 | 5 | 165.26 | 117.55 | 6 | 258.72 | 183.94 | 5 | 304.97 | 188.31 |
| 20 | 5 | 377.81 | 184.18 | 6 | 544.25 | 284.43 | 5 | 630.83 | 306.53 |
| 21 | 5 | 818.89 | 294.13 | 6 | 1177.47 | 438.94 | 5 | 1388.28 | 509.28 |
| 22 | 5 | 1665.64 | 468.80 | 6 | 2510.49 | 693.17 | 5 | 4484.46 | 790.72 |
| 23 | 5 | 3296.92 | 758.48 | 6 | 7401.08 | 1036.17 | 5 | 8818.85 | 1231.31 |
| 24 | 5 | 6129.03 | 1154.84 | 6 | 68628.94 | 6115.76 | 6 | 82959.59 | 8752.08 |

Table 1: $m = 2n$, Case A: Uniform quadratic coefficients in different fields.

**Acknowledgement.** The authors would like to thank Kai-Min Chung for his valuable comments. Feng-Hao Liu is supported by National Science Foundation for partial support under grant CNS-0347661 and CNS-0831293. Yun-Ju Huang and Bo-Yin Yang thank the Taiwan National Science Council and the Academia Sinica for partial support under grant NSC-100-2218-E-001-002 and the AS Career Award.

$q = 257$:

| $\alpha$ | 0.33 | | | 0.5 | | | 1 | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.04 | 10.73 | 4 | 0.04 | 10.72 | 4 | 0.04 | 10.72 |
| 11 | 4 | 0.08 | 10.79 | 4 | 0.09 | 10.78 | 4 | 0.09 | 10.78 |
| 12 | 4 | 0.17 | 10.87 | 4 | 0.18 | 10.84 | 4 | 0.19 | 10.84 |
| 13 | 4 | 0.39 | 12.01 | 4 | 0.39 | 11.97 | 4 | 0.40 | 11.97 |
| 14 | 4 | 0.81 | 13.45 | 4 | 0.79 | 13.30 | 4 | 0.81 | 13.30 |
| 15 | 4 | 1.57 | 15.87 | 4 | 1.58 | 15.84 | 4 | 1.61 | 15.84 |
| 16 | 4 | 5.16 | 19.25 | 4 | 5.18 | 19.25 | 4 | 5.29 | 19.25 |
| 17 | 4 | 8.64 | 23.93 | 4 | 8.50 | 23.88 | 4 | 8.62 | 23.88 |
| 18 | 5 | 96.43 | 96.28 | 5 | 96.25 | 96.62 | 5 | 97.21 | 96.62 |
| 19 | 5 | 227.97 | 163.66 | 5 | 227.02 | 164.15 | 5 | 229.34 | 164.28 |
| 20 | 5 | 529.64 | 264.56 | 5 | 523.05 | 266.69 | 5 | 526.53 | 266.63 |
| 21 | 5 | 1098.82 | 449.59 | 5 | 1113.43 | 451.03 | 5 | 1118.36 | 451.99 |
| 22 | 5 | 2214.86 | 660.96 | 5 | 2240.20 | 663.56 | 5 | 2251.87 | 664.66 |

$q \simeq n^3$:

| $\alpha$ | 0.33 | | | 0.5 | | | 1 | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.04 | 10.73 | 4 | 0.04 | 10.72 | 4 | 0.04 | 10.72 |
| 11 | 4 | 0.07 | 10.79 | 4 | 0.09 | 10.78 | 4 | 0.10 | 10.78 |
| 12 | 4 | 0.17 | 10.88 | 4 | 0.19 | 10.84 | 4 | 0.19 | 10.84 |
| 13 | 4 | 0.37 | 11.98 | 4 | 0.39 | 11.97 | 4 | 0.40 | 11.97 |
| 14 | 4 | 0.76 | 13.41 | 4 | 0.79 | 13.30 | 4 | 0.81 | 13.30 |
| 15 | 4 | 1.56 | 15.81 | 4 | 1.58 | 15.84 | 4 | 1.61 | 15.84 |
| 16 | 4 | 5.21 | 19.25 | 4 | 5.22 | 19.25 | 4 | 5.31 | 19.25 |
| 17 | 4 | 8.50 | 23.91 | 4 | 8.51 | 23.88 | 4 | 8.63 | 23.88 |
| 18 | 5 | 95.89 | 95.69 | 5 | 96.36 | 96.62 | 5 | 97.52 | 96.97 |
| 19 | 5 | 227.84 | 163.37 | 5 | 227.77 | 164.15 | 5 | 230.02 | 164.28 |
| 20 | 5 | 525.33 | 264.68 | 5 | 525.47 | 266.63 | 5 | 528.16 | 267.53 |
| 21 | 5 | 1099.79 | 449.93 | 5 | 1117.60 | 451.99 | 5 | 1121.32 | 452.75 |
| 22 | 5 | 2224.06 | 662.26 | 5 | 2249.15 | 664.66 | 5 | 2257.76 | 665.84 |

$q \simeq n^8$:

| $\alpha$ | 0.33 | | | 0.5 | | | 1 | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.23 | 10.56 | 4 | 0.29 | 10.56 | 4 | 0.31 | 10.56 |
| 11 | 4 | 0.72 | 10.56 | 4 | 0.84 | 10.56 | 4 | 0.87 | 10.56 |
| 12 | 4 | 1.94 | 10.56 | 4 | 2.17 | 11.59 | 4 | 2.23 | 11.59 |
| 13 | 4 | 4.77 | 11.75 | 4 | 5.19 | 11.75 | 4 | 5.31 | 11.80 |
| 14 | 4 | 32.03 | 18.93 | 4 | 30.67 | 19.33 | 4 | 30.66 | 19.27 |
| 15 | 4 | 84.80 | 35.41 | 4 | 80.79 | 39.12 | 4 | 80.83 | 39.06 |
| 16 | 4 | 351.28 | 67.17 | 4 | 309.33 | 67.12 | 4 | 311.81 | 67.12 |
| 17 | 4 | 713.00 | 115.78 | 4 | 580.91 | 115.56 | 4 | 586.40 | 115.75 |
| 18 | 5 | 12146.62 | 229.81 | 5 | 9036.60 | 230.56 | 5 | 9055.74 | 230.81 |
| 19 | 5 | 30324.51 | 455.72 | 5 | 20520.21 | 456.75 | 5 | 20667.89 | 456.75 |
| 20 | 5 | 78325.85 | 922.58 | 5 | 63441.66 | 923.93 | 5 | 59827.03 | 924.56 |
| 21 | 5 | 154844.15 | 1681.13 | 5 | 153899.89 | 1682.44 | 5 | 154433.97 | 1683.81 |
| 22 | 5 | 512341.23 | 3180.26 | 5 | 499117.57 | 3182.25 | 5 | 505281.53 | 3184.93 |

Table 2: $m = 2n$, Case B: Solution $\in \{1, -1, 0\}$ and different $q$'s with quadratic coefficients in different Gaussian distribution.

| $n$ | Deg | Time | Mem |
|---|---|---|---|
| 10 | 6 | 0.05 | 10.75 |
| 11 | 6 | 0.11 | 10.78 |
| 12 | 6 | 0.23 | 11.09 |
| 13 | 6 | 0.45 | 12.25 |
| 14 | 6 | 0.97 | 14.97 |
| 15 | 6 | 2.83 | 16.59 |
| 16 | 6 | 23.09 | 54.83 |
| 17 | 6 | 56.54 | 86.66 |
| 18 | 6 | 136.48 | 138.97 |
| 19 | 6 | 309.67 | 219.69 |
| 20 | 6 | 642.78 | 350.10 |
| 21 | 6 | 1405.60 | 564.16 |
| 22 | 6 | 3041.41 | 855.94 |
| 23 | 6 | 8944.66 | 1315.91 |
| 24 | 6 | 84963.55 | 8766.97 |

Table 3: $m = 2n$, Case C: Solution $\in \{2, -2, 1, -1, 0\}$ in $\mathbb{F}_q, q \simeq n^3$ with quadratic coefficients in Gaussian distribution $\alpha = 1$.

| $q$ | 3 | 257 | | $\simeq n^3$ | | $\simeq n^8$ | |
|---|---|---|---|---|---|---|---|
| $n$ | Time | Time | Ratio | Time | Ratio | Time | Ratio |
| 10 | 0.03 | 0.04 | 1.33 | 0.04 | 1.33 | 0.31 | 10.33 |
| 11 | 0.08 | 0.09 | 1.13 | 0.10 | 1.25 | 0.87 | 10.88 |
| 12 | 0.15 | 0.19 | 1.27 | 0.19 | 1.27 | 2.23 | 14.87 |
| 13 | 0.31 | 0.40 | 1.29 | 0.40 | 1.29 | 5.31 | 17.13 |
| 14 | 0.63 | 0.81 | 1.29 | 0.81 | 1.29 | 30.66 | 48.67 |
| 15 | 1.23 | 1.61 | 1.31 | 1.61 | 1.31 | 80.83 | 65.72 |
| 16 | 3.90 | 5.29 | 1.36 | 5.31 | 1.36 | 311.81 | 79.95 |
| 17 | 6.18 | 8.62 | 1.39 | 8.63 | 1.40 | 586.40 | 94.89 |
| 18 | 70.19 | 97.21 | 1.38 | 97.52 | 1.39 | 9,055.74 | 129.02 |
| 19 | 165.26 | 229.34 | 1.39 | 230.02 | 1.39 | 20,667.89 | 125.06 |
| 20 | 377.81 | 526.53 | 1.39 | 528.16 | 1.40 | 59,827.03 | 158.35 |
| 21 | 818.89 | 1,118.36 | 1.37 | 1,121.32 | 1.37 | 154,433.97 | 188.59 |
| 22 | 1,665.64 | 2,251.87 | 1.35 | 2,257.76 | 1.36 | 505,281.53 | 303.36 |

Table 4: Comparison with Case A and Case B ($\alpha = 1$) in $m = 2n$. The ratio is with respect to $\mathbb{F}_3$.

| $q$ | 5 | $\simeq n^3$ | |
|---|---|---|---|
| $n$ | Time | Time | Ratio |
| 10 | 0.05 | 0.05 | 1.00 |
| 11 | 0.10 | 0.11 | 1.10 |
| 12 | 0.20 | 0.23 | 1.15 |
| 13 | 0.39 | 0.45 | 1.15 |
| 14 | 0.84 | 0.97 | 1.15 |
| 15 | 2.41 | 2.83 | 1.17 |
| 16 | 19.47 | 23.09 | 1.19 |
| 17 | 47.38 | 56.54 | 1.19 |
| 18 | 114.30 | 136.48 | 1.19 |
| 19 | 258.72 | 309.67 | 1.20 |
| 20 | 544.25 | 642.78 | 1.18 |
| 21 | 1,177.47 | 1,405.60 | 1.19 |
| 22 | 2,510.49 | 3,041.41 | 1.21 |
| 23 | 7,401.08 | 8,944.66 | 1.21 |
| 24 | 68,628.94 | 84,963.55 | 1.24 |

Table 5: Comparison with Case A and Case C in $m = 2n$. The ratio is with respect to $\mathbb{F}_5$.

| $q$ | $\mathbb{F}_3$ | | | $\mathbb{F}_5$ | | | $\mathbb{F}_{257}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.01 | 9.53 | 6 | 0.03 | 10.72 | 3 | 0.02 | 9.53 |
| 12 | 4 | 0.03 | 9.53 | 6 | 0.12 | 11.09 | 4 | 0.10 | 10.81 |
| 14 | 4 | 0.32 | 12.39 | 6 | 0.39 | 13.00 | 4 | 0.40 | 13.47 |
| 16 | 4 | 1.11 | 15.02 | 6 | 1.43 | 16.94 | 4 | 1.49 | 17.28 |
| 18 | 4 | 3.85 | 21.12 | 6 | 4.68 | 26.88 | 4 | 5.14 | 26.25 |
| 20 | 4 | 11.88 | 35.00 | 6 | 23.51 | 44.94 | 4 | 27.49 | 46.60 |
| 22 | 4 | 53.65 | 61.25 | 5 | 1177.84 | 359.00 | 5 | 463.27 | 349.19 |
| 24 | 5 | 1277.55 | 571.21 | 6 | 5253.19 | 907.68 | 5 | 2289.08 | 1044.09 |
| 26 | 5 | 5458.47 | 1477.68 | 6 | 7798.39 | 2008.94 | 5 | 8898.08 | 2352.13 |
| 28 | 5 | 19398.69 | 3219.76 | 6 | 25802.27 | 4806.74 | 5 | 29501.02 | 4966.22 |
| 30 | 5 | 59976.91 | 6649.03 | 6 | 83591.07 | 11500.36 | 5 | 96003.79 | 12057.78 |

Table 6: $m = 2.5n$, Case A: Uniform quadratic coefficients in different fields.

$q = 257$:

| $\alpha$ | 0.33 | | | 0.5 | | | 1 | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.01 | 9.53 | 4 | 0.01 | 9.53 | 4 | 0.01 | 9.53 |
| 12 | 4 | 0.03 | 10.56 | 4 | 0.03 | 10.56 | 4 | 0.03 | 10.56 |
| 14 | 4 | 0.36 | 12.50 | 4 | 0.38 | 12.47 | 4 | 0.38 | 13.06 |
| 16 | 4 | 1.35 | 17.00 | 4 | 1.40 | 17.00 | 4 | 1.48 | 17.00 |
| 18 | 4 | 4.75 | 25.26 | 4 | 4.84 | 26.28 | 4 | 4.90 | 26.28 |
| 20 | 4 | 15.02 | 44.28 | 4 | 15.30 | 44.46 | 4 | 15.42 | 45.31 |
| 22 | 4 | 72.20 | 82.91 | 4 | 73.67 | 82.91 | 4 | 74.03 | 82.91 |

$q \simeq n^3$:

| $\alpha$ | 0.33 | | | 0.5 | | | 1 | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.01 | 9.53 | 4 | 0.02 | 9.53 | 3 | 0.01 | 9.53 |
| 12 | 4 | 0.03 | 10.56 | 4 | 0.04 | 10.56 | 4 | 0.03 | 10.56 |
| 14 | 4 | 0.36 | 12.47 | 4 | 0.37 | 12.47 | 4 | 0.39 | 13.12 |
| 16 | 4 | 1.34 | 17.01 | 4 | 1.39 | 17.00 | 4 | 1.49 | 17.00 |
| 18 | 4 | 4.73 | 25.28 | 4 | 4.88 | 26.28 | 4 | 4.92 | 26.28 |
| 20 | 4 | 15.33 | 44.42 | 4 | 15.41 | 44.63 | 4 | 15.44 | 45.31 |
| 22 | 4 | 72.43 | 82.91 | 4 | 73.72 | 82.91 | 4 | 74.29 | 83.94 |

$q \simeq n^8$:

| $\alpha$ | 0.33 | | | 0.5 | | | 1 | | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | Deg | Time | Mem | Deg | Time | Mem | Deg | Time | Mem |
| 10 | 4 | 0.02 | 9.53 | 4 | 0.02 | 9.53 | 4 | 0.02 | 9.53 |
| 12 | 4 | 0.07 | 9.53 | 4 | 0.09 | 10.22 | 4 | 0.09 | 10.56 |
| 14 | 4 | 18.83 | 15.45 | 4 | 19.02 | 14.72 | 4 | 20.01 | 14.64 |
| 16 | 4 | 131.31 | 31.91 | 4 | 134.38 | 32.78 | 4 | 134.39 | 32.94 |
| 18 | 4 | 639.94 | 93.37 | 4 | 642.45 | 93.69 | 4 | 640.00 | 94.38 |
| 20 | 4 | 2479.73 | 266.88 | 4 | 2540.31 | 267.57 | 4 | 2536.36 | 267.91 |
| 22 | 4 | 13572.34 | 625.22 | 4 | 13570.33 | 626.25 | 4 | 13544.13 | 627.62 |

Table 7: $m = 2.5n$, Case B: Solution $\in \{1, -1, 0\}$ and different $q$'s with quadratic coefficients in different Gaussian distribution.

| $n$ | Deg | Time | Mem |
|---|---|---|---|
| 10 | 6 | 0.02 | 10.72 |
| 12 | 6 | 0.10 | 11.09 |
| 14 | 5 | 0.43 | 14.03 |
| 16 | 6 | 1.56 | 19.03 |
| 18 | 6 | 5.26 | 28.97 |
| 20 | 6 | 27.02 | 51.12 |
| 22 | 6 | 1612.33 | 431.87 |
| 24 | 6 | 2341.27 | 1182.21 |
| 26 | 6 | 9142.07 | 2580.98 |
| 28 | 6 | 30187.27 | 5365.15 |
| 30 | 6 | 97189.53 | 12671.21 |

Table 8: $m = 2.5n$, Case C: Solution $\in \{2, -2, 1, -1, 0\}$ in $\mathbb{F}_q, q \simeq n^3$ with quadratic coefficients in Gaussian distribution $\alpha = 1$.

| $q$ | 3 | 257 | | $\simeq n^3$ | | $\simeq n^8$ | |
|---|---|---|---|---|---|---|---|
| $n$ | Time | Time | Ratio | Time | Ratio | Time | Ratio |
| 10 | 0.01 | 0.01 | 1.00 | 0.01 | 1.00 | 0.02 | 2.00 |
| 12 | 0.03 | 0.03 | 1.00 | 0.03 | 1.00 | 0.09 | 3.00 |
| 14 | 0.32 | 0.38 | 1.19 | 0.39 | 1.22 | 20.01 | 62.53 |
| 16 | 1.11 | 1.48 | 1.33 | 1.49 | 1.34 | 134.39 | 121.07 |
| 18 | 3.85 | 4.90 | 1.27 | 4.92 | 1.28 | 640.00 | 166.23 |
| 20 | 11.88 | 15.42 | 1.30 | 15.44 | 1.30 | 2,536.36 | 213.50 |
| 22 | 53.65 | 74.03 | 1.38 | 74.29 | 1.38 | 13,544.13 | 252.45 |

Table 9: Comparison with Case A and Case B ($\alpha = 1$) in $m = 2.5n$. The ratio is with respect to $\mathbb{F}_3$.

| $q$ | 5 | $\simeq n^3$ | |
|---|---|---|---|
| $n$ | Time | Time | Ratio |
| 10 | 0.03 | 0.02 | 0.67 |
| 12 | 0.12 | 0.10 | 0.83 |
| 14 | 0.39 | 0.43 | 1.10 |
| 16 | 1.43 | 1.56 | 1.09 |
| 18 | 4.68 | 5.26 | 1.12 |
| 20 | 23.51 | 27.02 | 1.15 |
| 22 | 1,177.84 | 1,612.33 | 1.37 |
| 24 | 5,253.19 | 2,341.27 | 0.45 |
| 26 | 7,798.39 | 9,142.07 | 1.17 |
| 28 | 25,802.27 | 30,187.27 | 1.17 |
| 30 | 83,591.07 | 97,189.53 | 1.16 |

Table 10: Comparison with Case A and Case C in $m = 2.5n$. The ratio is with respect to $\mathbb{F}_5$.

| $q$ | $\mathbb{F}_3$ | | $\mathbb{F}_5$ | | $\mathbb{F}_{257}$ | |
|---|---|---|---|---|---|---|
| $n$ | w/ solution | w/o solution | w/ solution | w/o solution | w/ solution | w/o solution |
| 10 | 0.03 | 0.04 | 0.05 | 0.03 | 0.04 | 0.04 |
| 11 | 0.08 | 0.07 | 0.10 | 0.08 | 0.09 | 0.09 |
| 12 | 0.15 | 0.15 | 0.20 | 0.18 | 0.20 | 0.20 |
| 13 | 0.31 | 0.30 | 0.39 | 0.37 | 0.42 | 0.42 |
| 14 | 0.63 | 0.63 | 0.84 | 0.80 | 0.92 | 0.93 |
| 15 | 1.23 | 1.24 | 2.41 | 2.34 | 2.76 | 2.75 |
| 16 | 3.90 | 3.88 | 19.47 | 19.34 | 22.06 | 22.06 |
| 17 | 6.18 | 6.16 | 47.38 | 47.54 | 55.36 | 55.34 |
| 18 | 70.19 | 70.07 | 114.30 | 113.94 | 134.08 | 133.82 |
| 19 | 165.26 | 164.66 | 258.72 | 258.75 | 304.97 | 304.84 |
| 20 | 377.81 | 378.12 | 544.25 | 543.29 | 630.83 | 630.96 |
| 21 | 818.89 | 823.16 | 1177.47 | 1180.77 | 1388.28 | 1402.32 |
| 22 | 1665.64 | 1660.39 | 2510.49 | 2542.00 | 4484.46 | 4509.49 |
| 23 | 3296.92 | 3333.50 | 7401.08 | 7352.91 | 8818.85 | 8785.89 |
| 24 | 6129.03 | 6121.40 | 68628.94 | 68666.17 | 82959.59 | 82938.45 |

Table 11: Comparison of Time(s) between system with and without solutions in Case A.

| $q$ | $\mathbb{F}_{257}$ | | $q \simeq n^3$ | | $q \simeq n^8$ | |
|---|---|---|---|---|---|---|
| $n$ | w/ solution | w/o solution | w/ solution | w/o solution | w/ solution | w/o solution |
| 10 | 0.04 | 0.04 | 0.04 | 0.04 | 0.31 | 0.34 |
| 11 | 0.09 | 0.09 | 0.10 | 0.09 | 0.87 | 0.94 |
| 12 | 0.19 | 0.20 | 0.19 | 0.20 | 2.23 | 2.45 |
| 13 | 0.40 | 0.41 | 0.40 | 0.42 | 5.31 | 5.60 |
| 14 | 0.81 | 0.92 | 0.81 | 0.92 | 30.66 | 35.22 |
| 15 | 1.61 | 2.74 | 1.61 | 2.74 | 80.83 | 143.68 |
| 16 | 5.29 | 21.95 | 5.31 | 22.04 | 311.81 | 1675.94 |
| 17 | 8.62 | 55.14 | 8.63 | 55.31 | 586.40 | 4577.53 |
| 18 | 97.21 | 133.48 | 97.52 | 133.83 | 9055.74 | 11817.38 |
| 19 | 229.34 | 304.45 | 230.02 | 305.09 | 20667.89 | 31752.26 |
| 20 | 526.53 | 630.09 | 528.16 | 632.38 | 59827.03 | 82175.13 |
| 21 | 1118.36 | 1356.02 | 1121.32 | 1367.34 | 154433.97 | 207822.16 |
| 22 | 2251.87 | 4387.78 | 2257.76 | 4403.85 | 505281.53 | 1145428.86 |

Table 12: Comparison of Time(s) between system with and without solutions in Case B($\alpha = 1$).

## References

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In Schulman [Sch10], pages 171–180.

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293, 1997.

[AFI$^+$04] Gwénolé Ars, Jean-Charles Faugère, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita. Comparison between xl and gröbner basis algorithms. In *ASIACRYPT*, pages 338–353, 2004.

[BBS86] Lenore Blum, Manuel Blum, and Mike Shub. A simple unpredictable pseudo-random number generator. *SIAM J. Comput.*, 15(2):364–383, 1986.

[BERW08] Andrey Bogdanov, Thomas Eisenbarth, Andy Rupp, and Christopher Wolf. Time-area optimized public-key engines: Mq cryptosystems as replacement for elliptic curves? Lecture Notes in Computer Science, 2008.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[BFP11] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Cryptanalysis of multivariate and odd-characteristic hfe variants. In *Public Key Cryptography*, pages 441–458, 2011.

[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. Previously INRIA report RR-5049.

[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)*, pages 1–14, 2005.

[BGP06] Côme Berbain, Henri Gilbert, and Jacques Patarin. Quad: A practical stream cipher with provable security. In *EUROCRYPT*, pages 109–128, 2006.

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.

[BPS08] Oliver Billet, Jacques Patarin, and Yannick Seurin. Analysis of intermediate field systems. In *SCC*, 2008.

[CCC$^+$09] Anna Inn-Tung Chen, Ming-Shing Chen, Tien-Ren Chen, Chen-Mou Cheng, Jintai Ding, Eric Li-Hsiang Kuo, Frost Yu-Shuang Lee, and Bo-Yin Yang. Sse implementation of multivariate pkcs on modern x86 cpus. In *CHES*, pages 33–48, 2009.

[CCD$^+$08] Chia-Hsin Owen Chen, Ming-Shing Chen, Jintai Ding, Fabian Werner, and Bo-Yin Yang. Odd-char multivariate hidden field equations. Cryptology ePrint Archive, Report 2008/543, 2008.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *EUROCRYPT*, pages 392–407, 2000.

[CN11] Yuanmi Chen and Phong Q. Nguyen. Bkz 2.0: Better lattice security estimates. In *ASIACRYPT*, pages 1–20, 2011.

[CP02] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *ASIACRYPT*, pages 267–287, 2002.

[DDY$^+$08] Jintai Ding, Vivien Dubois, Bo-Yin Yang, Chia-Hsin Owen Chen, and Chen-Mou Cheng. Could sflash be repaired? In *ICALP (2)*, pages 691–701, 2008.

[DFS07] Vivien Dubois, Pierre-Alain Fouque, and Jacques Stern. Cryptanalysis of sflash with slightly modified parameters. In *EUROCRYPT*, pages 264–275, 2007.

[DFSS07] Vivien Dubois, Pierre-Alain Fouque, Adi Shamir, and Jacques Stern. Practical cryptanalysis of sflash. In *CRYPTO*, pages 1–12, 2007.

[DGK$^+$10] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *TCC*, pages 361–381, 2010.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 1976.

[Die04] Claus Diem. The xl-algorithm and a conjecture from commutative algebra. In *ASIACRYPT*, pages 323–337, 2004.

[Fau02] Jean Charles Faugère. A new efficient algorithm for computing grouml;bner bases without reduction to zero (f5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (hfe) cryptosystems using gröbner bases. In *CRYPTO*, pages 44–60, 2003.

[FSS07] Reza Rezaeian Farashahi, Berry Schoenmakers, and Andrey Sidorenko. Efficient pseudorandom generators based on the ddh assumption. In *Public Key Cryptography*, pages 426–441, 2007.

[FY80] Aviezri S. Fraenkel and Yaacov Yesha. Complexity of solving algebraic equations. *Inf. Process. Lett.*, 10(4/5):178–179, 1980.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32, 1989.

[Gol01] Oded Goldreich. *Foundations of Cryptography. Basic tools.* Cambridge University Press, 2001.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[GRS95] Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries: The highly noisy case. In *FOCS*, pages 294–303, 1995.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[Hol06] Thomas Holenstein. Pseudorandom generators from one-way functions: A simple construction for any hardness. In *TCC*, pages 443–461, 2006.

[HRV10] Iftach Haitner, Omer Reingold, and Salil P. Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. In Schulman [Sch10], pages 437–446.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Structure in Complexity Theory Conference*, pages 134–147, 1995.

[Kal86] Burton S. Kaliski. A pseudo-random bit generator based on elliptic logarithms. In *CRYPTO*, pages 84–103, 1986.

[LLY08] Feng-Hao Liu, Chi-Jen Lu, and Bo-Yin Yang. Secure prngs from specialized polynomial maps over any gf(q). In *PQCrypto*, pages 181–202, 2008.

[Mac02] Francis. S. Macaulay. On some formulae in elimination. *Proceedings of the London Mathematical Society*, 1902.

[McE78] Robert McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN Progress Report*, 1978.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynominal-tuples for efficient signature-verification and message-encryption. In *EUROCRYPT*, pages 419–453, 1988.

[Pat95] Jacques Patarin. Cryptoanalysis of the matsumoto and imai public key scheme of eurocrypt'88. In *CRYPTO*, pages 248–261, 1995.

[Pat96] Jacques Patarin. Asymmetric cryptography with a hidden monomial. In *CRYPTO*, pages 45–60, 1996.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *STOC*, pages 333–342. ACM, 2009.

[PG97] Jacques Patarin and Louis Goubin. Asymmetric cryptography with s-boxes. In *ICICS*, pages 369–380, 1997.

[PGC98] Jacques Patarin, Louis Goubin, and Nicolas Courtois. $C^{*}_{-+}$ and hm: Variations around two schemes of t. matsumoto and h. imai. In *ASIACRYPT*, pages 35–49, 1998.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571, 2008.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

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

[Sch10] Leonard J. Schulman, editor. *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010.* ACM, 2010.

[Vad11] Salil P. Vadhan. Pseudorandomness. *Book draft*, 2011.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, pages 80–91, 1982.

[YC04] Bo-Yin Yang and Jiun-Ming Chen. All in the xl family: Theory and practice. In *ICISC*, pages 67–86, 2004.

[YCBC07] Bo-Yin Yang, Chia-Hsin Owen Chen, Daniel J. Bernstein, and Jiun-Ming Chen. Analysis of quad. In *FSE*, pages 290–308, 2007.

[YCC04] Bo-Yin Yang, Jiun-Ming Chen, and Nicolas Courtois. On asymptotic security estimates in XL and Gröbner bases-related algebraic cryptanalysis. In *ICICS 2004*, volume 3269, pages 401–413. Springer, Oct. 2004.