An efficient FHE proposal based on the hardness of solving systems of nonlinear multivariate equations (II)

Gérald Gavin

Laboratory ERIC

University of Lyon, France

Email: gavin@univ-lyon1.fr

Abstract. We propose a general framework to develop fully homomorphic encryption schemes (FHE) without using Gentry's technique. Initially, a private-key cryptosystem is built over $\mathbb{Z}_n$ (n being an RSA modulus). An encryption of $x \in \mathbb{Z}_n$ is a randomly chosen vector e such that $\Phi(e) = x$ where Φ is a secret multivariate polynomial. This private-key cryptosystem is not homomorphic in the sense that the vector sum is not a homomorphic operator. Non-linear homomorphic operators are then developed. The security relies on the difficulty of solving systems of nonlinear equations (which is a $\mathcal{NP}$-complete problem). While the security of our scheme has not been reduced to a provably hard instance of this problem, its security is globally investigated.

1 Introduction

The theoretical problem of constructing a fully homomorphic encryption scheme (FHE) supporting arbitrary functions f, was only recently solved by the breakthrough work of Gentry [3]. More recently, further fully homomorphic schemes were presented [7],[8],[1],[4] following Gentry's framework. The underlying tool behind all these schemes is the use of Euclidean lattices, which have previously proved powerful for devising many cryptographic primitives. A central aspect of Gentry's fully homomorphic scheme (and the subsequent schemes) is the ciphertext refreshing Recrypt operation. Even if many improvements have been made, this operation remains very costly [6], [5].

In [2], authors have presented a general framework to develop FHE without using the Gentry's technique. They first proposed a very simple private-key cryptosystem where a ciphertext is a vector e whose components are in $\mathbb{Z}_n$, n being an RSA modulus chosen at random. Given a secret multivariate polynomial Φ, an encryption of $x \in \mathbb{Z}_n$ is a vector e chosen at random such that $\Phi(e) = x$. In order to resist to a CPA attacker, the number of monomials of Φ should not be polynomial (otherwise the cryptosystem can be broken by solving a polynomial-size linear system). In order to get polynomial-time encryptions and decryptions, Φ should be written in a compact form, e.g. a factored or semi-factored form. By construction, the generic cryptosystem described above is not homomorphic in the sense that the vector sum is not a homomorphic operator. This is a *sine qua non* condition for overcoming Gentry's machinery. Indeed, as a ciphertext e is a vector, it is always possible to write it as a linear combination of other known ciphertexts. Thus, if the vector sum is a homomorphic operator, the cryptosystem is not secure at all. So, in order to use the vector sum as a homomorphic operator, noise should be injected into the encryptions as is done in all existing FHE. To overcome this, the authors propose developing *ad hoc* nonlinear homomorphic operators. The public key contains these operators and public encryptions while the secret key contains the multivariate polynomial Φ.

Our contribution. Our construction is strongly inspired by [2] where security was related to the difficulty of solving nonlinear equations in $\mathbb{Z}_n$. While the underlying ideas are the same, the construction proposed in this paper is simpler, more natural and more efficient. For concreteness, we consider the same private-key cryptosystem (see Section 2) and the homomorphic operators are still built with operators $\mathcal{Q}$ (see Section 4). Thus, the proof of Proposition 8 can be found (without any modification) in Appendix

C of [2]. This result provides a formal framework for the cryptanalysis by restricting the set of possible attacks. The main modifications with respect to [2] are provided in the construction of the homomorphic operators. The construction is no longer probabilistic[1] and is much more natural, allowing us to prove Lemma 2 and proposition 9. These results strongly suggest the non-existence of attacks by linearization in a relaxed but natural setting. The security in a real life setting is discussed in Section 6.3. but we have not provided formal results. The FHE presented in this paper is extremely simple, and potentially very efficient compared to other existing FHE.

2 A basic private-key cryptosystem

Let $m, \delta \in \mathbb{N}^*$ and n be an RSA modulus. All the computations of this paper will be done in $\mathbb{Z}_n$.

- *The set of all square m-by-m matrices over $\mathbb{Z}_n$ is denoted by $\mathbb{Z}_n^{m\times m}$.*
- *Throughout this paper, a vector $\vec{w} = \begin{pmatrix} w_1 \\ ... \\ w_m \end{pmatrix}$ can be also denoted by w or $(w_1, ..., w_m)$.*
- *Given $a \in \mathbb{Z}_n$ and two vectors w and w' of $\mathbb{Z}_n^m$:*
  - $w.w' = w_1 w_1' + ... + w_m w_m'$ *denotes the inner product of these vectors.*
  - $w \times w' = (w_1 w_1', ..., w_m w_m')$.
  - $wa = (w_1 a, w_2 ..., w_m)$.
- *A vector* B *is said to be basic if* B *is a δ-vector, i.e. $(b_1, ..., b_\delta) \in \mathbb{Z}_n^\delta$ and if*

$$\prod_{i=1}^{\delta} b_i = 1$$

  Throughout this paper, basic vectors will be denoted with (small) capital letters.
- *Let $w_1, ..., w_t$ be t vectors of size m, $(w_1, ..., w_t)$ denotes the concatenation of these vectors, i.e. $(w_1, ..., w_t) = (w_{11}, ..., w_{1m}, ..., w_{t1}, ..., w_{tm})$.*
- *Given a vector w and a matrix S, $|w|_S = Sw$. Note that $|w|_S$ could be denoted by $|w|$ when S is implicitly known.*

First, we define a private-key cryptosystem where the plaintext space is $\mathbb{Z}_n$ and where the secret key contains ϑ randomly chosen invertible matrices S_z of $\mathbb{Z}_n^{2\kappa\delta\times 2\kappa\delta}$. For $\kappa = 1$, a valid encryption e of x is composed of ϑ vectors $c_1, ..., c_\vartheta$ defined by

$$c_z = S_z^{-1}\ (\mathrm{A}_z x_z, \mathrm{B}_z)$$

where $\mathrm{A}_z, \mathrm{B}_z$ are randomly chosen basic vectors and the x_z are randomly chosen values satisfying $x_1 + ... + x_\vartheta = x$. A decryption consists of evaluating a δ-degree multivariate polynomial Φ,

$$\Phi(e) = \sum_{z=1}^{\vartheta} \prod_{i=1}^{\delta} s_{zi}.c_z = x$$

where s_{zi} denotes the i^{th} row of S_z. One should notice that the expanded representation of Φ is exponential-size provided $\delta = \Theta(\lambda)$: this is fundamental in the security analysis of the scheme. One can remark that the

[1] except for the choice of the associated matrices used to build operators $\mathcal{Q}$.

basic vectors B_z are not useful yet. They can be regarded as a stock of randomness useful for homomorphic operators to generate new encryptions. The role of the parameter ϑ will be explained in Section 6. We let the reader see why the scheme cannot be semantically secure with $\vartheta = 1$ (an attacker could easily decide if an encryption encrypts 0 or not). The parameter κ is artificially introduced in order to provide symmetry properties, which are fundamental in the proof of Proposition 8.

Definition 1. *Let λ be a security parameter. The functions KeyGen1, Encrypt1, Decrypt1 are defined as follows:*

1. *KeyGen1(λ). Let $\eta, \kappa, \delta, \vartheta$ be positive integers indexed by λ. Let n be a η-bit RSA modulus chosen at random and $(S_z)_{z=1,...,\vartheta}$ be ϑ invertible matrices of $\mathbb{Z}_n^{2\kappa\delta\times 2\kappa\delta}$ chosen at random. The i^{th} row of S_z is denoted by s_{zi}. For any $l \in \{1, ..., \kappa\}$, $\Phi_l : \left(\mathbb{Z}_n^{2\kappa\delta}\right)^{\theta} \to \mathbb{Z}_n$ denotes the δ-degree multivariate polynomial defined by $\Phi_l(w_1, ..., w_\vartheta) = \sum_{z=1}^{\vartheta} \quad \prod_{i\in I_l} s_{zi}.w_z$ with $I_l = \{2(l-1)\delta + 1, ..., 2(l-1)\delta + \delta\}$. Output*
$$K = \{(S_z)_{z=1,...,\vartheta}\}$$
2. *Encrypt1($K, x \in \mathbb{Z}_n$). Randomly choose $2\kappa\vartheta$ basic vectors[2] $(\mathrm{A}_{zl}, \mathrm{B}_{zl})_{(z,l)\in\{1,...,\vartheta\}\times\{1,...,\kappa\}}$ and $\kappa\vartheta$ values $(x_{zl})_{(z,l)\in\{1,...,\vartheta\}\times\{1,...,\kappa\}}$ belonging to $\mathbb{Z}_n$ such that for all $l = 1, ..., \kappa$, $x_{1l} + ... + x_{\vartheta l} = x$. Let $(c_z)_{z=1,...,\vartheta}$ be the ϑ vectors defined by:*
$$|c_z|_{S_z} = (\mathrm{A}_{z1}x_{z1}, \mathrm{B}_{z1}, \mathrm{A}_{z2}x_{z2}, \mathrm{B}_{z2}, ..., \mathrm{A}_{z\kappa}x_{z\kappa}, \mathrm{B}_{z\kappa})$$
Output $e = (c_1, ..., c_\vartheta)$.
3. *Decrypt1($K, e \in \left(\mathbb{Z}_n^{2\kappa\delta}\right)^{\vartheta}$. Choose $l \in \{1, ..., \kappa\}$ arbitrarily and output*
$$x = \Phi_l(e)$$

3 Operators $\mathcal{Q}$

The operators $\mathcal{Q}$ are the main tool of this paper. The homomorphic operators only consist of applying a polynomial number of such operators. Let $\kappa, m \in \mathbb{N}^*$ and S, S', S'' be three invertible matrices of $\mathbb{Z}_n^{\kappa m\times\kappa m}$. The i^{th} row of S, S', S'' is respectively denoted by s_i, s'_i, s''_i. An operator $\mathcal{Q}$ inputs two vectors (or only one, see Remark 1) w', w'' and outputs a vector w without revealing S, S', S'' such that each component of $|w|_S$ is a two-degree polynomial defined over $|w'|_{S'}$ and $|w''|_{S''}$.

Definition 2. *(Operators $\mathcal{Q}$). A κ-symmetric family of polynomials $p_1, ..., p_{\kappa m} : \mathbb{Z}_n^{\kappa m} \times \mathbb{Z}_n^{\kappa m} \to \mathbb{Z}_n^{\kappa m}$ with respect to S, S', S'' is a family of 2-degree polynomials defined by*
$$\forall (i, l) \in \{1, ..., m\} \times \{0, ..., \kappa - 1\},\ p_{i+lm}(w', w'') = \sum_{j=1}^{\alpha_i} a_{ij} \left(s'_{u'_{ij}+lm}.w'\right) \left(s''_{u''_{ij}+lm}.w''\right)$$
where $\alpha_i \in \mathbb{N}^*$, $a_{ij} \in Z_n$, $u'_{ij}, u''_{ij} \in \{1, ..., m\}$.

The function QGen inputs S and a κ-symmetric family (with respect to S, S', S'') of polynomials $(p_i)_{i=1,...,\kappa m}$ and outputs the expanded representation of the polynomials $q_1, ..., q_{\kappa m}$ defined by
$$(q_1, \ldots, q_{\kappa m}) = S^{-1} (p_1, \ldots, p_{\kappa m})$$
The operator[3] $\mathcal{Q} \leftarrow$ QGen$(S, p_1, ..., p_{\kappa m})$ consists of evaluating the expanded representation of the polynomials q_i and outputting $\mathcal{Q}(w', w'') = (q_1(w', w''), ..., q_{\kappa m}(w', w''))$.

[2] Recall a basic vector is a δ-vector such that the product of its components is equal to 1.
[3] also denoted by QGen$(S, S', S'', (a_{ij}, u'_{ij}, u''_{ij})_{i=1,...,m;j=1,...,\alpha_i})$

Remark 1. In the construction of our FHE, several operators $\mathcal{Q}$ are one-operand operators, i.e. they input only one vector w'. The construction of such operators is exactly the same: it suffices to consider that $w'' = w'$ and $S'' = S'$. The only difference is that the number of monomials of the polynomials $p_i(w')$ and thus $q_i(w')$ is approximatively divided by 2 (the monomials $w'_i w''_j$ and $w'_j w''_i$ can be regrouped), i.e. QGen outputs a number of monomial coefficients approximatively divided by 2.

Let $\mathcal{Q} \leftarrow \mathsf{QGen}(S, S', S'', (a_{ij}, u'_{ij}, u''_{ij})_{i=1,...,m;j=1,...,\alpha_i})$ and $w \leftarrow \mathcal{Q}(w', w'')$. By denoting Sw (resp. $S'w'$, $S''w''$) by $|w|$ (resp. $|w'|$, $|w''|$),

$$|w|_{i+lm} \overset{\mathsf{def}}{=} \rho_{i+lm}(|w'|, |w''|) = \sum_{j=1}^{\alpha_i} a_{ij} |w'|_{u'_{ij}+lm} |w''|_{u''_{ij}+lm}$$

It should be noticed that the polynomial ρ_{i+lm} ($l > 0$) can be deduced from the polynomial ρ_i (this explains why it suffices to consider the case $\kappa = 1$ in our construction). Higher degree polynomials ρ_i could be considered but this would lead to very costly operators $\mathcal{Q}$: the running time of such operators is exponential in the degree of ρ_i.

Given a matrix $M \in \mathbb{Z}_n^{\kappa m \times \kappa m}$, we denote by $M^{[1]}$ the first m rows of M, $M^{[2]}$ the m next rows... and $M^{[\kappa]}$ the m last rows of M. Given a permutation σ of $\{1, ..., \kappa\}$, we denote by M_σ the matrix obtained by permuting the blocks $M^{[1]}, ..., M^{[\kappa]}$ according to σ. We easily check that

$$\mathsf{QGen}(S, S', S'', (a_{ij}, u'_{ij}, u''_{ij})_{i=1,...,m;j=1,...,\alpha_i}) = \mathsf{QGen}(S_\sigma, S'_\sigma, S''_\sigma, (a_{ij}, u'_{ij}, u''_{ij})_{i=1,...,m;j=1,...,\alpha_i})$$

These symmetry properties lead to privacy properties encapsulated in Proposition 8.

Proposition 1. *Let $(p_i)_{i=1,...,\kappa m}$ be a κ-symmetric family of polynomials. The computation of $\mathcal{Q} \leftarrow \mathsf{QGen}(S, p_1, ..., p_{\kappa m})$ requires $O(\kappa^4 m^4)$ modular multiplications and the computation of $w \leftarrow \mathcal{Q}(w', w'')$ requires $O(\kappa^3 m^3)$ modular multiplications.*

Proof. (Sketch.) The number of monomials of each p_i is $O(\kappa^2 m^2)$.
□

4 Homomorphic operators

In order to simplify notations, our construction will be presented for $\kappa = 1$: the extension to the general case $\kappa > 1$ is straightforward according to Definition 2.

Throughout this section, S, S', R will denote three arbitrary invertible matrices of $\mathbb{Z}_n^{2\delta \times 2\delta}$ and $w, w' \in \mathbb{Z}_n^{2\delta}$ will denote two vectors such that $|w|_S = (\mathrm{A}x, \mathrm{B})$ and $|w'|_{S'} = (\mathrm{A}'x', \mathrm{B}')$ where $x, x' \in \mathbb{Z}_n$ and $\mathrm{A}, \mathrm{B}, \mathrm{A}', \mathrm{B}'$ are basic vectors.

All the matrices considered in this section belong to $\mathbb{Z}_n^{2\delta \times 2\delta}$.

4.1 Overview

Let $e = (c_z)_{z=1,...,\vartheta}$ and $e' = (c'_z)_{z=1,...,\vartheta}$ be two encryptions of x and x'. We wish to develop a public algorithm which computes a valid encryption $e'' = (c''_z)_{z=1,...,\vartheta}$ of $x + x'$ or xx' only using operators $\mathcal{Q}$. Intuitively, the $\mathcal{Q}$ allow manipulating the components of $|c_z| = S_z c_z$ and $|c'_z| = S_z c'_z$ by computing 2-degree polynomials. By combining these operators, (almost) arbitrary polynomials can be computed. Thanks to the constraints introduced in $\mathsf{Encrypt1}$, it is possible to define the components of $|c''_z| = S_z c''_z$ as polynomials of the components of $|c_1|, ..., |c_\vartheta|$ and $|c'_1|, ..., |c'_\vartheta|$: it follows that it is possible to implement homomorphic operators by only applying operators $\mathcal{Q}$. In the next section, we propose a construction using $O(\vartheta^3)$ operators $\mathcal{Q}$. In order to simplify the presentation of our construction, several intermediate operators will be considered.

4.2 Operator Rand

This simple operator is fundamental in the security analysis of our scheme (see proof of Lemma 2).

Definition 3. *Let σ_1, σ_2 be two permutations of $\{1, ..., \delta\}$. The procedure $\mathsf{RandGen}(R, S, \sigma_1, \sigma_2)$ outputs the operator $\mathcal{Q} \leftarrow \mathsf{QGen}(R, p_1, ..., p_{2\delta})$ where $(p_i)_{i=1,...,2\delta} : \mathbb{Z}_n^{2\delta} \to \mathbb{Z}_n^{2\delta}$ are the polynomials defined by*

$$p_i(w) = \begin{cases} (s_i.w)\left(s_{\sigma_1(i)+\delta}.w\right) & \text{if } i \in \{1, ..., \delta\} \\ (s_i.w)\left(s_{\sigma_2(i-\delta)+\delta}.w\right) & \text{if } i \in \{\delta+1, ..., 2\delta\} \end{cases}$$

The operator $\mathsf{Rand} \leftarrow \mathsf{RandGen}(R, S, \sigma_1, \sigma_2)$ simply consists of applying $\mathcal{Q}$, i.e. $\mathsf{Rand}(w) = \mathcal{Q}(w)$.

Proposition 2. *Let $\mathsf{Rand} \leftarrow \mathsf{RandGen}(R, S, \sigma_1, \sigma_2)$ and $v \leftarrow \mathsf{Rand}(w)$. It is ensured that*

$$|v|_R = (\sigma_1(\mathrm{B}) \times \mathrm{A}x, \sigma_2(\mathrm{B}) \times \mathrm{B})$$

where $\sigma_i(\mathrm{B})$ is the basic vector obtained by permutating the components of B according to σ_i.

Proof. By definition of operators $\mathcal{Q}$.
□

4.3 Operator $\mathsf{Substitute}$

The operator $\mathsf{Substitute}$, for instance applied to the vector w, allows to *progressively* replace the basic vectors A by a basic vector only depending on B.

$$|w|_S = \begin{pmatrix} a_1 x \\ a_2 \\ a_3 \\ a_4 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} \; ; \; |u_1|_U = \begin{pmatrix} a_1 a_2 x \\ a_3 a_4 \\ b_1 b_2 \\ b_3 b_4 \\ b_1^2 \\ b_2^2 \\ b_3^2 \\ b_4^2 \end{pmatrix} \; ; \; |v_1|_S = \begin{pmatrix} b_1^2 a_1 a_2 x \\ b_2^2 a_3 a_4 \\ b_3^2 b_1 b_2 \\ b_3 b_4^3 \\ b_1^4 \\ b_2^4 \\ b_3^4 \\ b_4^4 \end{pmatrix} \; ; |u_2|_U = \begin{pmatrix} b_1^2 b_2^2 x \\ b_3^2 b_4^2 \\ b_1^4 b_2^4 \\ b_3^4 b_4^4 \\ b_1^8 \\ b_2^8 \\ b_3^8 \\ b_4^8 \end{pmatrix} \; ; |v_2|_S = \begin{pmatrix} b_1^{10} b_2^2 x \\ b_2^8 b_3^2 b_4^2 \\ b_3^8 b_1^4 b_2^4 \\ b_3^4 b_4^{12} \\ b_1^{16} \\ b_2^{16} \\ b_3^{16} \\ b_4^{16} \end{pmatrix}$$

Fig. 1. Simulation of the execution of $\mathsf{Substitute}(w)$ (with the toy parameter $\delta = 4$) where the vector v_2 is output. One should notice that $|v_2|_S = (\mathrm{C}x, \mathrm{D})$ where C and D are basic vectors only depending on B.

Definition 4. *Let S be an invertible matrix. The procedure $\mathsf{SubstituteGen}(S)$ consists of executing the two following issues:*

1. *Generate the operators $\mathcal{Q}$ and Rand as follows:*
   - *Choose at random an invertible matrix U.*
   - *$\mathsf{Rand} \leftarrow \mathsf{RandGen}(S, U, \mathsf{Id}, \mathsf{Id})$ (where Id is the identity).*
   - *$\mathcal{Q} \leftarrow \mathsf{QGen}(U, p_1, ..., p_{2\delta})$ where $(p_i)_{i=1,...,2\delta} : \mathbb{Z}_n^{2\delta} \to \mathbb{Z}_n^{2\delta}$ are the polynomials defined by*

$$p_i(w) = \begin{cases} (s_{2i-1}.w)\,(s_{2i}.w) & \text{if } i \in \{1, ..., \delta\} \\ (s_i.w)\,(s_i.w) & \text{if } i \in \{\delta+1, ..., 2\delta\} \end{cases}$$

2. *Output the operator* $\mathsf{Substitute} \leftarrow \mathsf{SubstituteGen}(S)$ *defined as follows:*

   $\underline{\mathsf{Substitute}(w):}$

   $v_0 \leftarrow w$
   for $i = 1$ *to* $\lceil \log_2 \delta \rceil$
   $u_i \leftarrow \mathcal{Q}(v_{i-1})$
   $v_i \leftarrow \mathsf{Rand}(u_i)$
   Output $v_{\lceil \log_2 \delta \rceil}$

Proposition 3. *Let* $\mathsf{Substitute} \leftarrow \mathsf{SubstituteGen}(S)$*. The vector* $v \leftarrow \mathsf{Substitute}(w)$ *satisfies*

$$|v|_S = (\mathrm{C}x, \mathrm{D})$$

where C *and* D *are basic vectors only depending of* B*.*

Proof. It suffices to check that at each step i, $|v_i|_S = (\mathrm{C}_i x, \mathrm{D}_i)$ where $\mathrm{C}_i, \mathrm{D}_i$ are basic vectors. We conclude by noticing that the basic vectors C_i and D_i only depend of B (and not of A) provided $i \geq \lceil \log_2 \delta \rceil$.
□

4.4 Operator Add

The operator Add is fundamental in the construction of both homomorphic operators. Roughly speaking, it allows to add the hidden values x, x' associated to the vectors w and w'.

Definition 5. *Let* $a, a' \in \mathbb{Z}_n$*. The procedure* $\mathsf{AddGen}(R, S, S', a, a')$ *consists of executing the two following issues:*

1. *Generate the operators* $\mathcal{Q}_1, \mathcal{Q}_2, \mathcal{Q}_3$ *and* $\mathsf{Substitute}$ *defined as follows*

   – *Randomly choose an invertible matrix* U*. Let* $(p_i)_{i=1,...,2\delta}$*,* $(p'_i)_{i=1,...,2\delta}$*,* $(p''_i)_{i=1,...,2\delta}$ *be polynomials* $\mathbb{Z}_n^{2\delta} \times \mathbb{Z}_n^{2\delta} \to \mathbb{Z}_n^{2\delta}$ *defined by*

   $$p_i(w, w') = \begin{cases} (s_i.w)\,(s'_{i+\delta}.w') & \text{if } i \in \{1, ..., \delta\} \\ (s_i.w)\,(s'_i.w') & \text{if } i \in \{\delta+1, ..., 2\delta\} \end{cases}$$

   $$p'_i(w, w') = \begin{cases} (s_{i+\delta}.w)\,(s'_i.w') & \text{if } i \in \{1, ..., \delta\} \\ (s_i.w)\,(s'_i.w') & \text{if } i \in \{\delta+1, ..., 2\delta\} \end{cases}$$

   $$p''_i(w, w') = \begin{cases} a\,(u_1.w)\,(u_{1+\delta}.w') + a'\,(u_{1+\delta}.w)\,(u_1.w') & \text{if } i = 1 \\ (u_i.w)\,(u_{i+\delta}.w') & \text{if } i \in \{2, ..., \delta\} \\ (u_i.w)\,(u_i.w') & \text{if } i \in \{\delta+1, ..., 2\delta\} \end{cases}$$

   – $\mathsf{Substitute} \leftarrow \mathsf{SubstituteGen}(U)$

   – $\mathcal{Q}_1 \leftarrow \mathsf{QGen}(U, p_1, ..., p_{2\delta})$, $\mathcal{Q}_2 \leftarrow \mathsf{QGen}(U, p'_1, ..., p'_{2\delta})$ *and* $\mathcal{Q}_3 \leftarrow \mathsf{QGen}(R, p''_1, ..., p''_{2\delta})$

2. *Output the operator* $\mathsf{Add} \leftarrow \mathsf{AddGen}(R, S, S', a, a')$ *defined by:*

   $\underline{\mathsf{Add}(w, w'):}$

   (a) $w_1 \leftarrow \mathcal{Q}_1(w, w')$ *and* $w'_1 \leftarrow \mathcal{Q}_2(w, w')$.

   (b) $w_2 \leftarrow \mathsf{Substitute}(w_1)$ *and* $w'_2 \leftarrow \mathsf{Substitute}(w'_1)$.

   (c) *Output* $v \leftarrow \mathcal{Q}_3(w_2, w'_2)$

Proposition 4. *Let* $\mathsf{Add} \leftarrow \mathsf{AddGen}(R, S, S', a, a')$*. The vector* $v \leftarrow \mathsf{Add}(w, w')$ *satisfies*

$$|v|_R = (\mathrm{C}(ax + a'x'), \mathrm{D})$$

where C*,* D *are basic vectors.*

Proof. By construction, $|w_1|_S = (\mathrm{A} \times \mathrm{B}'x, \mathrm{B} \times \mathrm{B}')$ and $|w_1|_S = (\mathrm{A}' \times \mathrm{B}x', \mathrm{B} \times \mathrm{B}')$. Consequently, thanks to $\mathsf{Substitute}$, $|w_2|_U = (\mathrm{H}x, \mathrm{G})$ and $|w'_2|_U = (\mathrm{H}x', \mathrm{G})$ where H and G are functions of $\mathrm{B} \times \mathrm{B}'$. It follows that $|v|_R = (\mathrm{H} \times \mathrm{G}(ax + a'x'), \mathrm{G} \times \mathrm{G})$.
□

4.5 Operator Mult

Definition 6. *The procedure* $\mathsf{MultGen}(R, S, S')$ *outputs the operators* $Q \leftarrow \mathsf{QGen}(R, p_1, ..., p_{2\delta})$ *defined by*

$$p_i(w, w') = (s_i.w)\left(s'_i.w'\right)$$

The operator $\mathsf{Mult} \leftarrow \mathsf{MultGen}(R, S, S')$ *consists of applying* $\mathcal{Q}$*, i.e.* $\mathsf{Mult}(w, w') = \mathcal{Q}(w, w')$*.*

Proposition 5. *Let* $\mathsf{Mult} \leftarrow \mathsf{MultGen(R,S,S')}$*. The vector* $v \leftarrow \mathsf{Mult}(w, w')$ *satisfies*

$$|v|_R = (\mathrm{A} \times \mathrm{A}'xx', \mathrm{B} \times \mathrm{B}')$$

Proof. Straightforward.
□

4.6 Homomorphic operators

Let $K = (S_z)_{z=1,...,\vartheta} \leftarrow \mathsf{KeyGen}(\lambda)$ and $e = (c_z)_{z=1,...,\vartheta}$, $e' = (c'_z)_{z=1,...,\vartheta}$ be two valid encryptions of x and x', i.e.

- $|c_z|_{S_z} = (\mathrm{A}_z x_z, \mathrm{B}_z)$
- $|c'_z|_{S_z} = (\mathrm{A}'_z x'_z, \mathrm{B}'_z)$

Each homomorphic operator can be represented by an $\mathsf{Add}/\mathsf{Mult}$ circuit (see Fig. 2): $\oplus$ requires $O(\vartheta^2)$ operators Add while $\odot$ requires $O(\vartheta^2)$ operators Mult and $O\left(\vartheta^3\right)$ operators Add.

Operator $\oplus$. This homomorphic operator can be built by only using operators Add (thus only operators $\mathcal{Q}$).

Definition 7. *The procedure* $\mathsf{OplusGen}(K)$ *consists of executing the two following issues:*

1. *Generate the operators* $Add_{zz'}$ *as follows:*
   - *Let* $(\sigma_z)_{z=1,...,\vartheta}$ *be* ϑ *arbitrary permutations of* $\{1, ..., 2\vartheta\}$ *s.t.* $\sigma_z(1) = 2z - 1$*.*
   - *Randomly choose a family* $(a_{zz'})_{z=1,...,\vartheta;z'=1,....,2\vartheta}$ *of elements of* $\mathbb{Z}_n$ *such that* $\sum_{z=1}^{\vartheta} a_{z\sigma_z^{-1}(z')} = 1$ *for all* $z' = 1, ..., 2\vartheta$*.*
   - *Randomly choose a family* $(R_{zz'})_{z=1,...,\vartheta;z'=2,....,2\vartheta-1}$ *of invertible matrices.*
   - *State* $R_{z1} = S_z$*,* $R_{z,2\vartheta} = S_z$ *,* $b_{z1} = a_{z1}$ *and* $b_{zz'=2,...,2\vartheta-1} = 1$ *for all* $z = 1, ..., \vartheta$*.*
   - $\mathsf{Add}_{zz'} \leftarrow \mathsf{AddGen}(R_{zz'}, R_{z,z'-1}, S_{\lfloor \frac{\sigma_z(z')+1}{2} \rfloor}, b_{z,z'-1}, a_{zz'})$ *for all* $z = 1, ..., \vartheta$ *and* $z' = 2, ..., 2\vartheta$*.*

2. *Output the operator $\oplus \leftarrow \mathsf{OplusGen}(K)$ defined as follows:*

$\underline{e \oplus e'}$

$\quad$ *for* $z = 1$ *to* ϑ

$\quad\quad w_{2z-1} \leftarrow c_z$ *and* $w_{2z} \leftarrow c'_z$

$\quad$ *for* $z = 1$ *to* ϑ

$\quad\quad c''_z \leftarrow w_{\sigma_z(1)}$

$\quad\quad$ *for* $z' = 2$ *to* 2ϑ

$\quad\quad\quad c''_z \leftarrow \mathsf{Add}_{zz'}\left(c''_z, w_{\sigma_z(z')}\right)$

$\quad$ *Output* $(c''_1, ..., c''_\vartheta)$

Proposition 6. *The operator $\oplus \leftarrow \mathsf{OplusGen}(K)$ inputs two valid encryptions e, e' of x and x' and outputs a valid encryption $(c''_1, ..., c''_z) = e \oplus e'$ of $x + x'$ satisfying*

$$|c''_z|_{S''_z} = (\mathrm{A}''_z co_z(x_1, ..., x_\vartheta, x'_1, ..., x'_\vartheta), \mathrm{B}''_z)$$

where co_z are linear combinations chosen at random[4] *in $\mathsf{OplusGen}(K)$ satisfying*

$$\sum_{z=1}^{\vartheta} co_z(x_1, ..., x_\vartheta, x'_1, ..., x'_\vartheta) = x + x'$$

Proof. (*Sketch*) Our construction ensures that $|c''_z|_{S''_z} = (\mathrm{A}''_z co_z(x_1, ..., x_\vartheta, x'_1, ..., x'_\vartheta), \mathrm{B}''_z)$. The equalities $\sum_{z=1}^{\vartheta} a_{z\sigma_z^{-1}(z')} = 1$ for all $z' = 1, ..., 2\vartheta$ ensures that $\sum_{z=1}^{\vartheta} co_z(x_1, ..., x_\vartheta, x'_1, ..., x'_\vartheta) = x + x'$.
$\square$

Operator $\odot$. This homomorphic operator can be built by only using operators Add and Mult (see Fig. 2). The operators Mult allow to build ϑ^2 vectors hiding the products $x_i x'_j$. The encryption $e \odot e'$ is obtained by *summing* these products with $\vartheta(\vartheta^2 - 1)$ operators Add.

Definition 8. *The procedure $\mathsf{OdotGen}(K)$ consists of executing the two following issues:*

1. *Generate the operators $\mathsf{Mult}_{zz'}$ and $\mathsf{Add}_{zz'}$ defined as follows:*
   - *Let $(\sigma_z)_{z=1,...,\vartheta}$ be ϑ arbitrary bijections from $\{1, ..., \vartheta^2\}$ into $\{1, ..., \vartheta\}^2$ such that $\sigma_z(1) = (z, 1)$.*
   - *Randomly choose a family $(a_{zz'})_{z=1,...,\vartheta; z'=1,....,\vartheta^2}$ of elements of Z_n such that $\sum_{z=1}^{\vartheta} a_{z\sigma_z^{-1}(z'z'')} = 1$ for all $(z', z'') \in \{1, ..., \vartheta\}^2$.*
   - *Randomly choose a family $(T_{zz'})_{(z,z') \in \{1,...,\vartheta\}^2}$ of invertible matrices*
   - *$\mathsf{Mult}_{zz'} \leftarrow \mathsf{MultGen}(T_{zz'}, S_z, S_{z'})$*
   - *Randomly choose a family $(R_{zz'})_{z=1,...,\vartheta; z'=2,....,\vartheta^2-1}$ of invertible matrices*
   - *State $R_{z1} = T_{z1}$, $R_{z\vartheta^2} = S_z$, $b_{z1} = a_{z1}$ and $b_{zz'=2,...,\vartheta^2} = 1$ for all $z = 1, ..., \vartheta$.*
   - *$\mathsf{Add}_{zz'} \leftarrow \mathsf{AddGen}(R_{zz'}, R_{z,z'-1}, T_{\sigma_z(z')}, b_{z,z'-1}, a_{zz'})$ for all $z = 1, ..., \vartheta$ and $z' = 2, ..., \vartheta^2$.*

2. *Output the operator $\odot \leftarrow \mathsf{OdotGen}(K)$ defined as follows:*

$\underline{e \odot e'}$

$\quad w_{zz'} \leftarrow \mathsf{Mult}_{zz'}(c_z, c'_{z'})$ *for all* $z, z' \in \{1, ..., \vartheta\}^2$

$\quad$ *for* $z = 1$ *to* ϑ

$\quad\quad c''_z \leftarrow w_{\sigma_z(1)}$

$\quad\quad$ *for* $z' = 2$ *to* ϑ^2

$\quad\quad\quad c''_z \leftarrow \mathsf{Add}_{zz'}\left(c''_z, w_{\sigma_z(z')}\right)$

$\quad$ *Output* $(c''_1, ..., c''_\vartheta)$.

[4] related to the choice of the values $a_{zz'}$

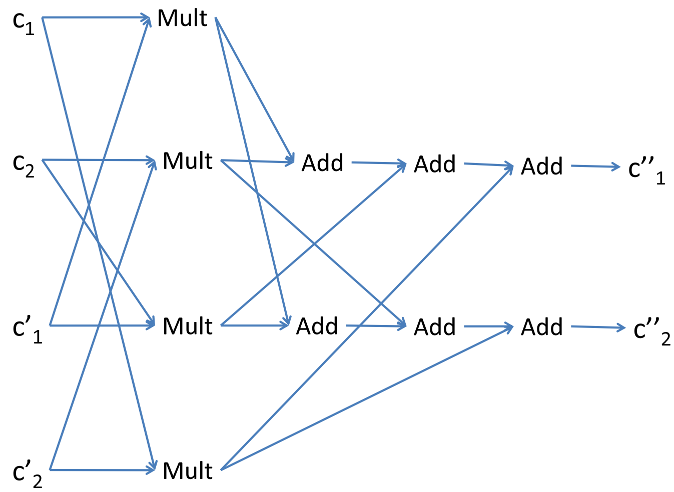

Fig. 2. Representation of the operator $\odot$ as a $\mathsf{Mult}/\mathsf{Add}$ circuit (for $\vartheta = 2$) where $(c''_1, c''_2) = (c_1, c_2) \odot (c'_1, c'_2)$.

Proposition 7. *The operator* $\odot \leftarrow \mathsf{OdotGen}(K)$ *inputs two valid encryptions* e, e' *and outputs a valid encryption* $(c''_1, ..., c''_z) = e \odot e'$ *of* xx' *satisfying*

$$|c''_z|_{S''_z} = (\mathrm{A}''_z co_z(x_1x'_1, ..., x_ix'_{i'}, ...x_\vartheta x'_\vartheta), \mathrm{B}''_z)$$

where co_z *are linear combinations chosen at random in* $\mathsf{OdotGen}(K)$ *satisfying*

$$\sum_{z=1}^{\vartheta} co_z(x_1x'_1, ..., x_ix'_{i'}, ...x_\vartheta x'_\vartheta) = xx'$$

Proof. Similar to the proof of proposition 6.
□

$\mathsf{OpGen}(K)$ outputs $\oplus \leftarrow \mathsf{OplusGen}(K)$ and $\odot \leftarrow \mathsf{OdotGen}(K)$. The whole number of operators $\mathcal{Q}$ involved in $\oplus$ and $\odot$ is $O(\vartheta^3 \log \delta)$.

5 The FHE

The private-key encryption scheme of Section 2 can be transformed in an FHE by publishing the homomorphic operators $\oplus, \odot$ and m encryptions $(e_v)_{v=1,...,m}$ of public values $x_v \in \mathbb{Z}_n$: for instance $x_v = 2^v \bmod n$.

Definition 9. *Let* λ *be a security parameter.*

- $\mathsf{KeyGen}(\lambda)$. *Let* $K = \{(S_z)_{z=1,...,\vartheta}\} \leftarrow \mathsf{KeyGen1}(\lambda)$, $\{\oplus, \odot\} \leftarrow \mathsf{OpGen}(K)$ *and for all* $v = 1, ..., m$, $e_v \leftarrow \mathsf{Encrypt1}(K, x_v)$.
$$sk = \{(S_z)_{z=1,...,\vartheta}\} \ ; pk = \{\oplus, \odot, (e_v)_{v=1,...,m}\}$$
- $\mathsf{Evaluate}(C, e_1, ..., e_m)$. *To evaluate* $C(e_1, ..., e_m)$, *it suffices to compute each gate with the public homomorphic operators* $\oplus$ *and* $\odot$.
- $\mathsf{Encrypt}(pk, x \in \mathbb{Z}_n)$. *It consists of evaluating a secret circuit* C *over the encryptions* $(e_v)_{v=1,...,m}$ *such that* $x = C(x_1, ..., x_m)$, *i.e. output* $\mathsf{Evaluate}(C, e_1, ..., e_m)$

– *Decrypt*(sk, e). *Exactly follows* Decrypt1.

The internal randomness of KeyGen can be decomposed in three parts:

– The internal randomness of KeyGen1 and OpGen is called *structural randomness*. For concreteness, by invoking KeyGen1, KeyGen generates the ϑ invertible matrices $S_1, ..., S_\vartheta$ of sk. By invoking OpGen(K), KeyGen randomly generates $O(\vartheta^3)$ other intermediate invertible matrices denoted by $S_{\vartheta+1}, ..., S_\Upsilon$ and $O(\vartheta^3)$ values $a_{zz'}$ used to build $O(\vartheta^3)$ operators Add. In the following of this paper, the i^{th} row of S_u is denoted by s_{ui}.

– The internal randomness of Encrypt1 used to build the public encryptions $(e_v)_{v=1,...,m}$ can be decomposed into two independent randomnesses:

  - The first one satisfying multiplicative constraints, called *multiplicative randomness*, comes from the choice of the basic vectors $\mathrm{A}_{vzl}, \mathrm{B}_{vzl}$ for each encryption $e_v \in pk$.
  - The second one satisfying additive constraints, called *additive randomness*, comes from the choice of the values x_{vzl} for each encryption $e_v \in pk$.

In Section 6, we will see that the independence of these three sources of randomness is important in the security analysis of the FHE. We define the following sets of polynomials (indexed by structural randomness meaning that each monomial coefficient is a function of the coefficients of $(S_u)_{u=1,...,\Upsilon}$):

– SP: the set of multi-variate polynomials $\phi : \left(\mathbb{Z}_n^{2\kappa\delta}\right)^r \to \mathbb{Z}_n$ defined by

$$\phi(w_1, ..., w_r) = \prod_{t=1}^{\gamma} s_{u_t i_t}.w_{k_t}$$

where $\gamma, r \in \mathbb{N}^*$, $i_t \in \{1, ..., 2\kappa\delta\}$, $u_t \in \{1, ..., \Upsilon\}$ and $k_t \in \{1, ..., r\}$.

– SP^γ: the set of polynomials of SP of degree equal to γ, i.e.

$$\mathsf{SP}^\gamma = \{\phi \in \mathsf{SP} | \deg(\phi) = \gamma\}$$

Security naturally deals with these polynomials because they allow computing polynomials over the components of $|w_k|_{S_u}$. For instance, the decryption polynomials Φ_l (see Definition 1) are a sum of ϑ polynomials of SP^δ. A representation R_ϕ of an arbitrary polynomial ϕ is said to be effective if its storage is polynomial and if it allow to evaluating ϕ in polynomial time.

Proposition 8. *Let $\gamma \in \mathbb{N}^*$ such that γ is not a multiple of κ. Let $\phi \in \mathsf{SP}^\gamma$ and R_ϕ be an effective representation of ϕ. By assuming the hardness of factorization, recovering R_ϕ only given pk is difficult.*

Proof. Because the private-key cryptosystem is exactly the same as in [2] and the homomorphic operators are also built by only using operators $\mathcal{Q}$, the proof can be found in Appendix C of [2].
□

Corollary 1. *By assuming the hardness of the factorization, the secret matrices $(S_u)_{u=1,...,\Upsilon}$ cannot be polynomially recovered only given pk.*

The analysis of this proposition in [2] is entirely applicable (without any modification) here. Let us summarize it by assuming that $\delta = \Theta(\lambda)$ and $\kappa = \Theta(\lambda^{\epsilon>0})$. Proposition 8 seems *a priori* not sufficient to ensure security because the knowledge of the polynomials $\phi \notin \mathsf{SP}^\gamma$ could be used to break semantic security, e.g. $\phi = \Phi_1 + ... + \Phi_\kappa$ or $\phi = \Phi_1 ... \Phi_\kappa$ (see Definition 1). However, the expanded representation of these two polynomials is exponential-size and thus cannot be recovered. Besides, according to Proposition 8, it is difficult to find any of its natural effective representations, i.e. sum of products of *small* polynomials of SP. This analysis suggests that such polynomials (having an exponential-size expanded representation) cannot be recovered. But maybe polynomial-size polynomials ϕ (having polynomial numbers of monomials) could be used to break semantic security. Moreover, by considering the monomial coefficients of such polynomials as independent variables, they could be recovered by solving a linear system. Such attacks, called *attacks by linearization*, will be extensively studied in the next section.

6 Attacks by linearization

The public key pk can be naturally regarded as a system (Sys) of nonlinear equations. Proposition 8 tends to show that the resolution of (Sys) is quite intractable. However, this does not prevent our scheme against attacks by linearization. For instance, the most natural linearization attack consists of solving the linear system

$$\phi(e_i) = x_i$$

where $(e_i)_{i=1,...m}$ is a family of encryptions of $(x_i)_{i=1,...,m}$ and ϕ is a multivariate polynomial[5] of degree δ such that its monomial coefficients are the variables of the linear system. Provided m is sufficiently large, its resolution provides a linear combination ϕ^* of the decryption polynomials $(\Phi_l)_{l=1,...,\kappa}$. However, provided $\delta = \Theta(\lambda)$, this attack fails because the number of monomials of ϕ is exponential. Because of the introduction of homomorphic operators, new polynomial relations leading to new efficient attacks by linearization could appear.

6.1 General framework

In the following of the paper, Ω denotes the set of valid encryptions, i.e. the output space of Encrypt1[6]. Given $r \in \mathbb{N}^*$ and $e \in \Omega^r$, we naturally extend the function Decrypt by writing $\mathsf{Decrypt}(e) = (\mathsf{Decrypt}(e_1), ..., \mathsf{Decrypt}(e_r))$.

Let us consider an arbitrary efficient public procedure H_{pk} which inputs a polynomial-size tuple of encryptions $e \in \Omega^r$ and outputs a polynomial-size tuple $y \in \mathbb{Z}_n^m$, i.e. $y \leftarrow H_{pk}(e)$.

Definition 10. *(Efficient non-trivial linearization attack). Let $\phi : \mathbb{Z}_n^m \to \mathbb{Z}_n$ be a polynomial. Given a tuple $x \in \mathbb{Z}_n^r$, we define the subsets Ω^x, $\Omega(\phi)$ of Ω^r by:*

- $\Omega^x = \{e \in \Omega^r \mid \mathsf{Decrypt}(e) = x\}$
- $\Omega(\phi) = \{e \in \Omega^r \mid \phi(H_{pk}(e)) = 0\}$.

We say that there exists an efficient non-trivial linearization attack relative to ϕ, H_{pk} and x if

1. *The number of monomials of ϕ is polynomial*
2. $\frac{|\Omega^x \backslash \Omega(\phi)|}{|\Omega^x|}$ *is negligible*
3. $\frac{|\Omega^r \backslash \Omega(\phi)|}{|\Omega^r|}$ *is non negligible*

The two first properties ensure that the expanded representation of ϕ can be found by solving a linear system. The third property means that $\phi(H_{pk}(e)) = 0$ is not *trivially* satisfied, i.e., for any $e \in \Omega^r$. It could mean that $\phi(H_{pk}(e)) = 0$ is satisfied with higher probability whether e encrypts x rather than $x' \neq x$ giving an advantage to the attacker for distinguishing between $\mathsf{Decrypt}(e) = x$ and $\mathsf{Decrypt}(e) = x'$. If

[5] having the same monomials as the decryption polynomials Φ_l.

[6] Ω is the set of encryptions which can be output by Encrypt1. Decrypt1 can be modified in order to check whether e belongs to Ω before to decrypt.

this property is not satisfied, such advantages cannot be derived from ϕ and we say that the linearization attack is *trivial*.

Before to analyze our scheme against such attacks in real life setting, we will consider the natural relaxed setting where H_{pk} is constrained as follows:

Setting 1. *H_{pk} inputs $e \in \Omega^r$, computes new encryptions $e'_1, ..., e'_{r'}$ and outputs all the vectors considered in this computation ($e, e'_1, ..., e'_{r'}$ and all intermediate vectors output by operators $\mathcal{Q}$).*

For instance, H_{pk} could simply consists of computing $\odot$, i.e. $H_{pk}(e_1, e_2)$ outputs all the vectors considered in the computation of $e_1 \odot e_2$. The procedure H_{pk} can be represented by a $\mathsf{Add}/\mathsf{Mult}$ circuit $C_{H_{pk}}$.

6.2 Linearization attacks in Setting 1

In order to simplify our analysis (and the task of the attacker), throughout this section, n is assumed to be a large prime instead of an RSA modulus and we modify KeyGen as follows:

- According to Lemma 1 in [2], if there exists a linearization attack for $\kappa > 1$ then there exists a linearization attack for $\kappa = 1$. Thus, it suffices to consider the case $\kappa = 1$.
- The matrices S_u are assumed to be all equal to the same matrix S chosen at random. In particular, all the secret matrices $(S_z)_{z=1,...,\vartheta}$ of sk are equal to S. Given a vector w, Sw will be simply denoted by $|w|$ (instead of $|w|_S$).

Let $x = (x_1, ..., x_r) \in \mathbb{Z}_n^r$ and $e = (e_1, ..., e_r)$ uniformly drawn over Ω^x. Each encryption e_i is a tuple of vectors $(c_{iz})_{z=1,...,\vartheta}$ defined by $|c_{iz}| = (\mathrm{A}_{iz}x_{iz}, \mathrm{B}_{iz})$ where A_{iz}, B_{iz} are basic vectors and $x_{i1} + ... + x_{i\vartheta} = x_i$. The set of all the values x_{iz} is denoted by X and the set of the components of the basic vectors $\mathrm{A}_{iz}, \mathrm{B}_{iz}$ is denoted by C. According to $\mathsf{Encrypt1}$,

$$C \perp X$$

Let $y = (w_1, ..., w_t) \leftarrow H_{pk}(e)$ be the concatenation of the vectors outputs by H_{pk}. We define the auxiliary functions $\widetilde{H}_{pk}(e)$, $H^+_{pk}(e)$ and $H^*_{pk}(e)$ as follows:

- $\widetilde{H}_{pk}(e)$ outputs the tuple $\widetilde{y} = (|w_1|, ..., |w_t|)$.
- $H^+_{pk}(e)$ outputs the tuple $(p_k)_{k=1,...,2\delta t}$ defined[7] by

$$p_k = \begin{cases} \widetilde{y}_k ... \widetilde{y}_{k+\delta-1}, & \text{if } k \mod 2\delta = 1 \\ 1 & \text{Otherwise.} \end{cases}$$

- $H^*_{pk}(e)$ outputs the tuple $(\pi_k)_{k=1,...,2\delta t}$ where $\pi_k = \widetilde{y}_k / p_k$.

Lemma 1. *Let $y \leftarrow H_{pk}(e)$, $\widetilde{y} \leftarrow \widetilde{H}_{pk}(e)$, $p \leftarrow H^+_{pk}(e)$ and $\pi \leftarrow H^*_{pk}(e)$ and $\phi \in \mathsf{SP}$.*

1. *$\phi(y)$ is a product of $\deg \phi$ components of $\widetilde{y}$,*
2. *Each component of π is a product of elements of C and each component of p is a polynomial defined over X.*

Proof.

1. By definition of SP.
2. By induction on the size of $C_{H_{pk}}$.

□

Corollary 2. *If e is uniformly drawn over Ω^x,*

$$\pi \perp p$$

---

[7] or equivalently $p_k = \begin{cases} |w_{1+\lfloor (k-1)/2\delta \rfloor}|_1 ... |w_{1+\lfloor (k-1)/2\delta \rfloor}|_\delta, & \text{if } k \bmod 2\delta = 1 \\ 1 & \text{Otherwise.} \end{cases}$

Role of the parameter δ

Let us show that $\phi(y)$ depends on the multiplicative randomness for any "small" polynomial $\phi \in \mathsf{SP}$. In order to simplify our analysis (and the task of the attacker), we slightly modify $\mathsf{Encrypt1}$ such that the basic vectors A_{iz} and B_{iz} are all equal to the same basic vector $\mathrm{C} = (c_1, ..., c_\delta)$ chosen at random, i.e. $\mathrm{A}_{iz} = \mathrm{B}_{iz} = \mathrm{C}$. In other words, it is assumed that the encryptions $(e_1, ..., e_r)$ input in H_{pk} were built only using the basic vector C, i.e. $C \simeq \mathrm{C}$.

Lemma 2. *For any $d \in \mathbb{Z}$, the product $c_1^d ... c_\delta^d$ of components of C is said to be trivial*[8]*. Let $\pi \leftarrow H^*_{pk}(e)$. Any product $\pi_{k_1} ... \pi_{k_t}$ is a non-trivial product of components of C provided $t < \delta/4$.*

Proof. (*Sketch.*) Let us define the set $\Pi_{e',e \geq e'}$ as follows:

$$\Pi_{e',e} = \{c_{i_0}^{e'} c_{i_1} ... c_{i_{e-e'}} \mid i_0, ..., i_{e-e'} \in \{1, ..., \delta\}\}$$

By construction, $\pi_k \in \Pi_{e_k, e_k}$ (e_k being a power of 2) provided $k - 1 \mod 2\delta \geq \delta$. Moreover, thanks to the operator Rand, it is ensured that

$$\forall k = 1, ..., m \quad \pi_k \in \Pi_{e_k/4, e_k} \tag{1}$$

Let $\beta = \pi_{k_1} ... \pi_{k_t}$ be an arbitrary product of t components of π and $e^* = \max_{i=1,...,t}(e_{k_i})$. According to (1), β is a product $c_1^{a_1} ... c_\delta^{a_\delta}$ of components of C such that $\max_{i=1,...,\delta} a_i \geq e^*/4$. It follows that β should be a product of at least $\delta e^*/4$ components of C in order to be trivial, i.e.

$$\sum_{i=1}^{t} e_{k_i} \geq \delta e^*/4$$

As $te^* \geq \sum_{i=1}^{t} e_{k_i}$, we obtain $t \geq \delta/4$.
□

Remark 2. Let $\alpha = c_1^{i_1} ... c_\delta^{i_\delta}$ be an arbitrary non-trivial product of components of C assuming $i_1 \leq i_2 \leq ... \leq i_\delta$. It follows that $\alpha = c_2^{i_2 - i_1} ... c_\delta^{i_\delta - i_1}$ is independent of C (i.e. $\alpha = 1$) if and only if $i_2 - i_1 \equiv ... \equiv i_\delta - i_1 \equiv 0 \mod \lambda(n)/2$. As n is randomly chosen, α is independent of C with negligible probability.

Role of the parameter ϑ

Each component of $p = (p_k)_{k=1,...,m} \leftarrow H^+_{pk}(e)$ is a polynomial function defined over X. These polynomials depend on the values $a_{zz'}$ chosen in $\mathsf{OplusGen}(K)$ and $\mathsf{OdotGen}(K)$. These values are chosen at random ensuring some relations of the form

$$\sum_{i=1}^{\vartheta} p_{k_i} = f(x_1, ..., x_r)$$

where f is a multivariate polynomial. Is there a non-trivial polynomial relation involving $t < \vartheta$ components of p? The following conjecture says that such relation can occur but only with negligible probability over the choice of the values $a_{zz'}$.

Conjecture 1. Let $t < \vartheta$, $k \in \{1, ..., m\}^t$, $x = (x_1, ..., x_r) \in \mathbb{Z}_n^r$ be arbitrarily chosen. Let $\nu : \mathbb{Z}_n^t \to \mathbb{Z}_n$ be a polynomial-size polynomial such that $\deg \nu = O(\lambda)$*. The set $\Omega(\nu)$ refers to the set of $e \in \Omega^r$ such that $\nu(p_{k_1}, ..., p_{k_t}) = 0$ where $p \leftarrow H^+_{pk}(e)$. We say ν is not trivial relatively to k, x if*

[8] It is equal to 1 because C is assumed to be a basic vector.

– $|\Omega^x \setminus \Omega(\nu)|/|\Omega^x|$ *is negligible*
– $|\Omega^r \setminus \Omega(\nu)|/|\Omega^r|$ *is non negligible.*

There exists a non-trivial polynomial ν relatively to k, x with negligible probability over the choice of $(pk, sk) \leftarrow \mathsf{KeyGen}(\lambda)$.

Roughly speaking, this conjecture says that if $\nu(p_{k_1}, ..., p_{k_t}) = 0$ is satisfied by (almost) all tuples $e \in \Omega^x$ then this relation is trivial[9] in the sense that $\nu(p_{k_1}, ..., p_{k_t}) = 0$ is also satisfied by (almost) all tuples $e \in \Omega^r$. This conjecture is discussed in Appendix A for the case $\vartheta = 2$.

Put it all together

We prove here the main result of this section. This result proves the non-existence of linearization attacks by assuming that vectors input in Add or Mult are randomized. We will then discuss this assumption in order to see how to overcome it.

Proposition 9. *Let us consider an oracle $\mathcal{O}_R$ which inputs a vector w such that $|w|_S = (\mathrm{A}x, \mathrm{B})$, generates two basic vectors C, D at random and outputs v defined by $|v|_S = (\mathrm{C}x, \mathrm{D})$. Let us modify Mult or Add in the following way: each input vector w is first randomized by $\mathcal{O}_R$. Let $x \in \mathbb{Z}_n^r$ and ϕ be an arbitrary polynomial-size linear combination of polynomials of SP, i.e. $\phi = a_1\varphi_1 + ... + a_\gamma\varphi_\gamma$ with $a_i \in \mathbb{Z}_n^*$ and $\varphi_i \in \mathsf{SP}$. Assuming Conjecture 1, there exists an efficient non-trivial linearization attack relative to ϕ, x with negligible probability provided $\delta = \Theta(\lambda)$ and $\vartheta = \Theta(\lambda)$.*

Proof. Given a tuple of encryptions $e \in \Omega^r$, $y = (w_1, ..., w_t) \leftarrow H_{pk}(e)$, $\widetilde{y} = (\widetilde{y}_k)_{k=1,...,m=2\delta t} \leftarrow \widetilde{H}_{pk}(e)$, $\pi \leftarrow H_{pk}(e)^*$ and $p \leftarrow H_{pk}(e)^+$. Let us partition the set $\{1, ..., m\}$ with the disjoint subsets $K_u = \{k \in \{1, ..., m\} | y_k$ is computed at the u^{th} node of $C_{pk}\}$.

For sake of simplicity, we assume that $\vartheta = \delta = \Theta(\lambda)$ and that the degree of each polynomial φ_i is equal to d, i.e. $\deg \varphi_i = d$ for all $i = 1, ..., \gamma$. According to Lemma 1, there exists a family $(k_{ij})_{i=1,...,\gamma; j=1,...,d}$ of elements of $\{1, ..., m\}$ such that

$$\phi(y) = \sum_{i=1}^{\gamma} a_i \prod_{j=1}^{d} \widetilde{y}_{k_{ij}} = \sum_{i=1}^{\gamma} a_i \prod_{j=1}^{d} \pi_{k_{ij}} p_{k_{ij}} \tag{2}$$

Let us assume the existence of an efficient linearization attack relative to ϕ and $x \in \mathbb{Z}_n^r$. If $d \geq \delta/5$, the linearization attack is not efficient (because the number of monomials is exponential) implying that $d < \delta/5$. According to Lemma 1, each product $\pi_{k_{i1}} ... \pi_{k_{id}}$ is a product of elements of C. At first, we assume that these products are equal[10], i.e.

$$\pi_{k_{11}} ... \pi_{k_{1d}} = ... = \pi_{k_{\gamma 1}} ... \pi_{k_{\gamma d}} \tag{3}$$

Thus, according to Definition 10, for each $e \in \Omega^x$ (except maybe for a negligible subset of Ω^x), p satisfies the following equation,

$$\nu(p) \stackrel{\mathsf{def}}{=} \sum_{i=1}^{\gamma} a_i \prod_{j=1}^{d} p_{k_{ij}} = 0 \tag{4}$$

Note that ν is a polynomial-size polynomial with $\deg \nu = d = O(\lambda)$

According to Lemma 2, any sub-product of $\pi_{k_{i1}}, ..., \pi_{k_{id}}$ is a non-trivial product of elements of C. By introducing $\mathcal{O}_R$ in our construction, Lemma 2 holds *a fortiori*.

[9] For instance $\nu(p_{k_1}, ..., p_{k_t}) = p_{k_1} - p_{k_1}$.
[10] for all the choices of C.

As vectors are assumed to be randomized by $\mathcal{O}_R$ before to be input in Add or Mult, it is ensured that $\pi_k \perp \pi_{k'}$ provided k and k' do not belong to the same set K_u. Because of (3) and Lemma 2,

$$\exists (i,j) \text{ s.t. } k_{ij} \in K_u \Rightarrow \forall i \in \{1,...,\gamma\}, \exists j \in \{1,...,d\} \text{ s.t. } k_{ij} \in K_u \tag{5}$$

Our construction ensures that each set $P_u = \{p_k | k \in K_u\}$ contains at most 4 elements[11]. Thus, according to (4) and (5), the set $P = \{p_{k_{ij}} \mid i = 1,...,\gamma; j = 1,...,d\}$ contains at most $t \leq 4d \leq 4\delta/5 < \delta = \vartheta$ elements denoted by $p_{k_1},...,p_{k_t}$. It follows that $\nu(p)$ only depends on $p_{k_1},...,p_{k_t}$.

According to Corollary 2, $p \perp \pi$. Thus, it suffices that $\nu(p) = 0$ to ensure that $e \in \Omega(\phi)$. Consequently, according to Conjecture 1, $|\Omega(\phi)|/|\Omega^r| \geq 1-\epsilon(\lambda)$ where $\epsilon(\lambda)$ is negligible. It implies that this linearization attack is trivial.

Now, let us consider the general case where the equality (2) is no longer satisfied. Let R be the equivalence relation defined over $\{1,...,\gamma\}$ defined by iRi' if and only if $\pi_{k_{i1}}...\pi_{k_{id}} = \pi_{k_{i'1}}...\pi_{k_{i'd}}$ (for all the choices of C). The equivalence classes of R are denoted by $I_1, I_2, ..., I_\tau$. By regrouping the products $\pi_{k_{i1}}...\pi_{k_{id}}$ which are equal, ϕ can be written as a sum of τ polynomials ϕ_v, i.e. $\phi = \phi_1 + ... + \phi_\tau$ with

$$\phi(y) = \sum_{v=1}^{\tau} \left(\sum_{i \in I_v} a_i \prod_{j=1}^{d} p_{k_{ij}} \right) \Pi_v$$

where Π_v is a product of components of C. Fixing p, $\phi(y)$ can be seen as a multivariate polynomial defined over C (the Π_v being the monomials). As $\phi(y) = 0$ for all encryptions $e \in \Omega^x$, this polynomial is identically equal to 0 (at the beginning of this section, n was assumed to be a large prime). It implies that for each $e \in \Omega^x$, $p \leftarrow H^+_{pk}(e)$ satisfies

$$\forall v = 1,...,\tau, \ \sum_{i \in I_v} a_i \prod_{j=1}^{d} p_{k_{ij}} = 0$$

Thus, the previous analysis can be done τ times leading to τ sets $\Omega_v(\phi_v)$ satisfying $|\Omega(\phi_v)|/|\Omega^r| \geq 1-\epsilon_v(\lambda)$. As $\bigcap_{v=1}^{\tau} \Omega_v(\phi_v) \subseteq \Omega(\phi)$ and $|\bigcap_{v=1}^{\tau} \Omega_v(\phi_v)|/|\Omega^r| \geq 1-(\epsilon_1+...+\epsilon_\tau)(\lambda)$, the linearization attack is still trivial.

□

Proposition 9 does not prove the non-existence of efficient non-trivial linearization attacks. However, it allows us to argue in favor of the fact that even if such an attack exist, a polynomial attacker can recover it with negligible probability.

- *Removing* $\mathcal{O}_R$. The use of $\mathcal{O}_R$ ensures that (3) ⇒ (5). Roughly speaking, assertion (5) implies that the number of nodes of $C_{H_{pk}}$ involved[12] in the linearization attack should be small, i.e. $O(d)$. But, Conjecture 1 ensures that any linearization dealing with less than $\vartheta/3$ nodes of $C_{H_{pk}}$ is surely trivial. Thus, if $d = \circ(\vartheta)$, the linearization attack is trivial provided $\vartheta = \Theta(\lambda)$. Conversely, if $d = \Omega(\lambda)$ the linearization attack is not efficient.

What happens if $\mathcal{O}_R$ is removed? Can we still guarantee that the number of nodes of $C_{H_{pk}}$ involved in the linearization attack is small? To see this, let us consider two arbitrary sets of vectors W_1, W_2 computed in *distant* sets of nodes of $C_{H_{pk}}$. We denote Π_1 and Π_2 the sets of components of $\pi \leftarrow H_{pk}(e)^*$ associated to respectively the sets W_1 and W_2. The use of $\mathcal{O}_R$ ensures that Π_1 and Π_2 are independent.

[11] P_u contains $1, q_u(X), r_u(X), q_u(X) + r_u(X)$ where q_u, r_u are polynomials, if Add is computed at the u^{th} node of C_{pk} or $1, q_u(X), r_u(X), q_u(X)r_u(X)$ if it is Mult.

[12] A node u is said to be involved if the linearization attack deals with at least one vector w computed at node u.

To prove Proposition 9 without $\mathcal{O}_R$, one needs a result establishing that *small* products of components of Π_1 cannot be equal to *small* products of components of Π_2. This seems intuitively true. Nevertheless, to get a formal result, one should quantify to notion of *distant sets* and *small products*.

Moreover, one can imagine several ways to emulate $\mathcal{O}_R$ keeping Lemma 2 true. For instance, the operator Rand (where the permutations σ and σ' would be randomly chosen) could be applying *several* times on each vector w input in Add or Mult. In our opinion, it should be possible modify our construction in this sense in order to extend Proposition 9 without considering $\mathcal{O}_R$.

• *Arbitrary* ϕ. Proposition 9 assumes that ϕ is a polynomial-size linear combination of polynomials of SP. As the polynomials $\phi_{ki}(w_1, ..., w_r) = w_{ki}$ can be written as a polynomial-size linear combination of polynomials of SP^1, any polynomial ϕ of degree $d = O(1)$ can be written as a polynomial-size linear combination of polynomials of $\cup_{t=0}^{d}\mathsf{SP}^t$. However there are polynomial-size polynomials ϕ (e.g. $\phi(y) = y_1 ... y_\delta$) of degree $d \neq O(1)$ which cannot. Proposition 9 does not exclude the existence of linearization dealing with such polynomials. However, provided $\delta = \Theta(\lambda)$, the number of $d-$degree monomials defined over the components of at least one vector w computed by H_{pk}, is not polynomial. It implies that some of them should be eliminated to get a polynomial attack. Intuitively, these monomials play a symmetric role and we do see how an attacker could choose some of them *a priori*.

Conjecture 2. Assuming $\delta = \vartheta = \Theta(\lambda)$, an attacker can find an efficient linearization attack in Setting 1 with negligible probability (randomness being the internal randomness of KeyGen*).*

6.3 Linearization attacks in real-life setting

In this (short) section, H_{pk} is not constrained anymore: it can use pk arbitrarily. Here, we list possible weaknesses of our scheme. While deeper investigations have been done, we did not get formal results. This section can be seen as a guideline for deeper cryptanalysis.

• In Setting 1, it was assumed that H_{pk} uses the public operators $\mathcal{Q}$ as required by $\oplus$ and $\odot$. Can an attacker get any advantage by inputting arbitrary vectors in operators $\mathcal{Q}$? For instance, it would be the case if $U \leftarrow S$ instead of being randomly chosen in $\mathsf{SubstituteGen}$. Indeed, in this case, the operator Rand could be removed from the construction (the loop would just consist of applying the operator $\mathcal{Q}$) leading to an efficient linearization attack from this[13]. As U and S are independently drawn, it seems irrelevant to input u_i in $\mathcal{Q}$. Roughly speaking, the information contained in u_i is associated to the matrix U while $\mathcal{Q}$ *works* on the information associated to S[14]. As each vector computed in $\oplus$ or $\odot$ is associated to a unique matrix, it can be assumed that vectors are not interchangeable in our construction.

• New operators $\mathcal{Q}$ could be polynomially derived from pk leading to efficient linearization attacks. Let us shortly argue against this (see Appendix B for a more detailed discussion about this):

- Proposition 8 ensures that it is not possible to recover the coefficients (or products of $\gamma < \kappa$ coefficients) of the matrices $(S_u)_{u=1,...,\Upsilon}$ involved in the public operators $\mathcal{Q}$.
- Randomness can be arbitrarily introduced in any public operator $\mathcal{Q}$ without altering the security analysis of the previous sections. This is detailed in Appendix K of [2]. By doing this, the system of equations derived from public operators $\mathcal{Q}$ becomes widely unknown.

7 Efficiency

The computation of an operator $\mathcal{Q}$ requires $O(\kappa^3\delta^3)$ multiplications in $\mathbb{Z}_n$. Moreover, $\oplus$ requires the application of $O(\vartheta^2 \log \delta)$ operators $\mathcal{Q}$ and $O(\vartheta^3 \log \delta)$ for $\odot$. Thus, denoting by $M(n)$ the runtime of

[13] We let the reader find it.

[14] More formally, it means that a linear perturbation is done over u_i before the nonlinear phase.

multiplications in $\mathbb{Z}_n$, the running time per addition gate is $O(\vartheta^2\kappa^3\delta^3 \log \delta M(n))$ and the running time per multiplication gate is $O(\vartheta^3\kappa^3\delta^3 \log \delta M(n))$. The running time of decryption is $O(\vartheta\kappa\delta^2 M(n))$. A ciphertext contains ϑ $2\kappa\delta$-vectors in $\mathbb{Z}_n$, implying that the ratio cipher size/plaintext size is equal to $2\kappa\vartheta\delta$. In terms of storage, the biggest part of the public key is the operator $\mathcal{Q}$, containing $O(\kappa^3\delta^3)$ elements of $\mathbb{Z}_n$, which leads to a space complexity in

$$O(|n|\vartheta^3\kappa^3\delta^3 \log \delta)$$

Attacks (in particular attacks by linearization) should be better quantified in order to propose instantiations of the parameters. Moreover, we think that the operators $\oplus$ and $\odot$ could be defined with respectively ϑ and ϑ^2 (instead of ϑ^2 and ϑ^3 operators Add) while remaining true Conjecture 1 and thus Proposition 9. The parameter κ was only introduced to prove Proposition 8. However, this parameter is not useful for protecting the scheme against attacks by linearization. Can we choose $\kappa = 1$? Similarly, n is assumed to be an RSA modulus. Can n be chosen prime? small prime?

8 Discussion and open questions

In this paper, a very simple FHE based on very simple tools was developed. Its security is linked to the difficulty of solving nonlinear systems of equations. By using arguments of symmetry, it was shown that the resolution of the system of equations (derived from pk) is intractable. However, it is not sufficient to ensure security against attacks by linearization. The main obstacle proving security consists of showing that all linear attacks are exponential. We argue in this sense but further investigations should be made. Moreover, improvements of our scheme deal with important open questions:

- The symmetry properties related to the parameter κ provide formal security guarantees but this parameter is not useful for protecting the scheme against attacks by linearization. Can this parameter be fixed to 1?
- the resolution of systems of nonlinear equations is $\mathcal{NP}$-complete in Z_n even if the factorization of n is known. Thus, one can wonder whether n can be chosen as a large prime? a small prime?

A positive answer to these questions would lead to an efficient FHE competitive with other classical (even not homomorphic) cryptosystems.

References

1. Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. Public key compression and modulus switching for fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 446–464, 2012.
2. Gérald Gavin. An efficient fhe based on the hardness of solving systems of non-linear multivariate equations. Cryptology ePrint Archive, Report 2013/262, 2013. http://eprint.iacr.org/.
3. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.
4. Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In *EUROCRYPT*, pages 465–482, 2012.
5. Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the aes circuit. In *CRYPTO*, pages 850–867, 2012.
6. Kristin Lauter, Michael Naehrig, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? *IACR Cryptology ePrint Archive*, 2011:405, 2011.
7. Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In *ASIACRYPT*, pages 377–394, 2010.
8. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010.

A What about Conjecture 1 ?

In this section, we wish testing Conjecture 1. In our experiments, H_{pk} inputs only one encryption $e = (c_z)_{z=1,\ldots,\vartheta}$ where $|c_z|_{S_z} = (\mathrm{A}_z x_z, \mathrm{B}_z)$. $H_{pk}(e)$ consists of computing new encryptions, e.g. $e \oplus e$, $e \odot e$, $(e \oplus e) \odot e$,....To validate Conjecture 1, it is required to show that there does not exist non-trivial polynomial relation between strictly less than ϑ components of $p \leftarrow H_{pk}(e)^+$. It is important to note that it is not required to really compute homomorphic operators $\oplus$ and $\odot$ to get $p = (p_k)_{k=1,\ldots,m}$: the components of p only depend of the values $a_{zz'}$ used in $\mathsf{OdotGen}$ and $\mathsf{OplusGen}$.

By renaming the values $a_{zz'}$ by $a, b, c, \ldots$, we list (by using Maple) the components of p expressed as polynomial functions of x_1 and $x = x_1 + x_2$. The components of p belong to this set

$$
\begin{aligned}
&\{1, x_1, ax_1 + b(x - x_1), ax_1 + b(x - x_1) + cx_1, ax_1 + b(x - x_1) + cx_1 + d(x - x_1), \\
&x - x_1, (1 - a)x_1 + (1 - b)(x - x_1), (1 - a)x_1 + (1 - b)(x - x_1) + (1 - c)x_1, \\
&(1 - a)x_1 + (1 - b)(x - x_1) + (1 - c)x_1 + (1 - d)(x - x_1), \\
&x_1^2, (x - x_1)^2, x_1(x - x_1), ex_1^2 + fx_1(x - x_1), ex_1^2 + fx_1(x - x_1) + g(x - x_1)(x - x_2), \\
&(a^2 - 2ab + ac - ad + b^2 - bc + bd - 4b)x_1 + (ab + ad + 2b - b^2 - bd)x, \ldots \\
&\}
\end{aligned}
$$

By checking *at the hand* each component of p, we see there does not exist constant components which only depend on x. Each component can take an exponential number of values. It means that there does not exit a small polynomial ν ensuring $\nu(p_k) = 0$ for any[15] $k = 1, \ldots, m$. It suggests that Conjecture 1 is true for $\vartheta = 2$.

B Informal discussion...

One could imagine that new operators $\mathcal{Q}$ can be polynomially derived from pk leading to efficient linearization attacks. Let us informally argue against this.

First, let us see that there does not any exist general method to achieve this, by considering the simple operator $\mathcal{Q} \leftarrow \mathsf{QGen}(S, p_1, \ldots, p_m)$ where S is an arbitrary invertible matrix of $\mathbb{Z}_n^{m \times m}$ $(m > 2)$ and $p_i(w') = s_i w' \times s_i w'$ for all $i = 1, \ldots, m$. Let $\sigma, \sigma' : \{1, \ldots, m\} \to \{1, \ldots, m\}$ be two arbitrary functions such that $\sigma \neq \mathsf{Id}$ or $\sigma' \neq \mathsf{Id}$ and $p'_i(w') = s_{\sigma(i)} w' \times s_{\sigma'(i)} w'$. By exploiting Proposition 1 of [2], one easily shows that recovering the operator $\mathcal{Q}' \leftarrow \mathsf{QGen}(S, p'_1, \ldots, p'_m)$ only given $\mathcal{Q}$ is difficult (assuming the hardness of factorization). Indeed, it suffices to note that the monomial coefficients of Q can be written as m-symmetric polynomials of the tuples $(s_1, \ldots, s_m)$ (where s_i is the i^{th} row of S) while the monomial coefficients of Q' are not m-symmetric.

In our construction, public operators $\mathcal{Q}$ encapsulate "more randomness" because they involve two or three invertible matrices chosen at random. This intuitively makes our problem more complex. Let $\mathcal{Q}_0$ be a public operator dealing with three matrices S, S', S'' chosen at random. Our construction ensures that there do not exist other operators $\mathcal{Q}$ dealing with the same triplet[16] of matrices S, S', S''. Each (public) monomial coefficient $q_{j_1 j_2 j_3}$ of $\mathcal{Q}_0$ is a sum of products of coefficients of $T = S^{-1}$, S' and S'', i.e.

$$
q_{j_1 j_2 j_3} = \sum_{i=1}^{2\kappa\delta} t_{j_1 i} s'_{u'_i j_2} s''_{u''_i j_3}
$$

[15] except if $p_k = 1$, but in this case, the relation is trivial.

[16] except the two operators $\mathcal{Q}$ considered in $\mathsf{Substitute}$. Modifications of $\mathsf{Substitute}$ can avoid this.

where u'_i and u''_i are parameters[17] belonging to $\{1, ..., 2\kappa\delta\}$. Let us focus on the possibility of recovering a new operator[18] $\mathcal{Q}'_0$. Fortunately, Proposition 8 ensures that it is not possible to recover the coefficients of S, S', S'', implying that $\mathcal{Q}'_0$ cannot be directly built. The construction of this operator requires computing the coefficients $q'_{j_1 j_2 j_3}$, which are sums of products of the coefficients of $T = S^{-1}$, S' and S''. Some of the involved products do not appear in any public value related to the public operators $\mathcal{Q}$ (because $\mathcal{Q}'_0 \neq \mathcal{Q}_0$ and the other operators $\mathcal{Q}$ do not deal with the same triplet of matrices S, S', S''). It follows that $q'_{j_1 j_2 j_3}$ cannot be obtained by computing linear combinations of these values. Moreover, it is difficult to imagine expressing $q'_{j_1 j_2 j_3}$ as polynomials (or ratios of polynomials) of these values. Nevertheless, *a priori*, nothing excludes public encryptions $(e_v)_{v=1,...,m}$ being able to build $\mathcal{Q}'_0$. However, these vectors depend on the randomness introduced in Encrypt1. This randomness is independent of the matrices $(S_u)_{u=1,...,\Upsilon}$ used to build the $\mathcal{Q}$. Efficient linearization attacks in Setting 1 would allow removing this randomness, giving values depending only on the coefficients of the matrices $(S_u)_{u=1,...,\Upsilon}$ involved in the public operators $\mathcal{Q}$. According to Conjecture 2, such attacks do not exist. This suggests that this randomness cannot be removed and thus public encryptions $(e_v)_{v=1,...,m}$ cannot be used to recover $q'_{j_1 j_2 j_3}$.

Furthermore, without altering the security analysis of the previous sections, randomness can be introduced in the choice of any public operator $\mathcal{Q}$, making the system of equations derived from operators $\mathcal{Q}$ widely unknown. The simplest way to achieve this consists of adding free (not involved in constraints) components $i = 2\kappa\delta+1, ...$ and choosing polynomials $(p_i)_{i=2\kappa\delta+1,...}$ (see Section 3) at random: an arbitrary number (each p_i provides $\Theta(\delta^2)$ new variables) of new variables[19] are introduced in the equations induced by each operator $\mathcal{Q}$. Another one is presented in detail in Appendix K of [2].

[17] To simplify, the coefficients α_i are assumed to be equal to 1 (see Definition 2).
[18] still assuming $\alpha_i = 1$ for all $i = 1, ..., 2\kappa\delta$.
[19] independent of pk