New ideas to build noise-free homomorphic cryptosystems

Gerald Gavin[1] and Sandrine Tainturier[2]

[1] Laboratory ERIC - University of Lyon
gerald.gavin@univ-lyon1.fr
[2] Adecco - Geneve
sandrine-tainturier@orange.fr

Abstract. We design a very simple private-key encryption scheme whose decryption function is a rational function. This scheme is not born naturally homomorphic. To get homomorphic properties, a nonlinear additive homomorphic operator is specifically developed. The security analysis is based on symmetry considerations and we prove some formal results under the factoring assumption. In particular, we prove IND-CPA security in the generic ring model. Even if our security proof is not complete, we think that it is convincing and that the technical tools considered in this paper are interesting by themselves. Moreover, the factoring assumption is just needed to ensure that solving nonlinear equations or finding non-null polynomials with many roots is difficult. Consequently, the ideas behind our construction could be re-used in rings satisfying these properties. As motivating perspectives, we then propose to develop a simple multiplicative operator. To achieve this, randomness is added in our construction giving hope to remove the factoring assumption in order to get a pure multivariate encryption scheme.

Keywords. *Homomorphic cryptosystem, Multivariate encryption scheme, Generic ring model.*

1 Introduction

The prospect of outsourcing an increasing amount of data storage and management to cloud services raises many new privacy concerns for individuals and businesses alike. The privacy concerns can be satisfactorily addressed if users encrypt the data they send to the cloud. If the encryption scheme is homomorphic, the cloud can still perform meaningful computations on the data, even though it is encrypted.

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 [Gen09]. More recently, further fully homomorphic schemes were presented [SS10],[vDGHV10],[CNT12],[GHS12a],[GSW13] 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 in one decade, this operation remains very costly [LNV11], [GHS12b], [DM15], [CGGI18]. Indeed, bootstrapped bit operations are still about one billion times slower than their plaintext equivalents (see [CGGI18]).

In this paper, we adopt another approach where a ciphertext is a vector $\boldsymbol{c}$ over $\mathbb{Z}_n$, n being an RSA modulus chosen at random. Given a secret multivariate rational function Φ_0/Φ'_0, an encryption of $x \in \mathbb{Z}_n$ is a vector $\boldsymbol{c}$ chosen at random ensuring that $\Phi_0/\Phi'_0(\boldsymbol{c}) = x$. Clearly, the expanded representations of Φ_0, Φ'_0 should not be polynomial-size (otherwise the CPA attacker could recover them by solving a polynomial-size linear system). In order to get polynomial-time encryptions and decryptions, Φ_0/Φ'_0 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 $\boldsymbol{c}$ is a vector, it is always possible to write it as a linear combination of other known ciphertexts. Thus, if the vector sum were a homomorphic operator, the cryptosystem would not be secure at all. This simple remark suffices to prove the weakness of the homomorphic cryptosystems presented in [XBY12], [KH12]. In order to use the vector sum as a homomorphic operator, noise should be injected into the encryptions as done in all existing FHE [Gen09],[BV11],[SS10],[vDGHV10],[CNT12],[GHS12a]. To get homomorphic properties, we develop *ad hoc* a nonlinear additively homomorphic operator Add and we obtain a noise-free additive encryption scheme.

The factoring assumption restricts the adversary's power providing hope to base the security of our scheme on this assumption. We prove a result based on symmetry (see Lemma 1) encapsulating the idea that it is not possible to extract roots of polynomials in $\mathbb{Z}_n$ intuitively meaning that a CPA attacker can only solve linear equations. For concreteness, Lemma 1 ensures that it cannot recover non-symmetric values only given symmetric values. By construction the CPA attacker has only access to symmetric values. Thus, it suffices to prove that breaking semantic security requires to recover non-symmetric values. Compact representations of Φ_0 or Φ'_0 deal with non-symmetric values implying that they cannot be recovered according to Lemma 1. However, $\Phi_0(\boldsymbol{c}) = 0$ provided $\boldsymbol{c}$ encrypts 0 implying that the expanded representation of Φ_0 could be recovered by solving a linear system. This kind of attacks will be called *attacks by linearization*. This attack fails by adjusting the parameters in order that Φ_0 has an exponential number of monomials. Nevertheless, the introduction of homomorphic operators may introduce new attacks by linearization. In section 5.3, we propose to formally define this class of attacks and we prove that such attacks do not exist against our scheme.

In Section 5.4, we propose a security analysis in the generic ring model [AM09], [JS09]. In this model, the power of the CPA attacker is restricted in the sense that it can only perform arithmetic operations. Recently, some results were shown in the generic ring model. For instance, it was shown that breaking the security of RSA in the generic ring model is as difficult as factoring [AM09]. An emblematic counterexample against security analysis in the generic ring model deals with Jacobi's symbol J_n. For concreteness, it was shown in [JS09] that computing J_n is difficult in the generic ring model while it is not in general. However, this result is neither surprising nor relevant because J_n is not a rational function[1]. Indeed, we can even show that $\Phi(x) = J_n(x)$ with probability smaller than $1/2$ provided Φ is a rational function and x uniform over $\mathbb{Z}_n^*$. Moreover, as far as we know, there does not exist any rational function provably difficult to compute in the generic ring model but not in general. Moreover, the analysis in the generic ring model excludes lattice-based attacks which *works outside* $\mathbb{Z}_n$. Nevertheless, all the considered random variables are uniform over $\mathbb{Z}_n$ contrarily to noise values considered in lattice-based cryptosystems.

We propose a general result reducing the generic IND-CPA security to algebraic conditions (Proposition 7). These results essentially come from a fundamental result (see Theorem 1) shown in [AM09] claiming that, under the factoring assumption, it is difficult to recover non-null polynomials having many roots. We then prove generic IND-CPA security (see Proposition 8).

Although we prove some results suggesting the security of our scheme, the security proof is not complete. Moreover the performance of our scheme is not competitive with respect to other existing additively homomorphic schemes (e.g. Paillier [Pai99], El Gamal [Elg85], Castagnos et al. [CL15]). So it is legitimate to question the usefulness of this paper. In our opinion, the underlying ideas of this paper are very promising and the proposed construction can be seen as a feasibility study. We see at least two motivating perspectives from this work. The principal one would be to build a multiplicative homomorphic operator. In Section 6, we propose a noise-free compact-FHE. The algebraic condition proposed for the homomorphic additive encryption remains valid. This condition could be exploited to get a formal security proof at least in the generic ring model. We propose a very short security analysis at least showing that our construction has a chance to be secure. A second motivating perspective would be to remove the factoring assumption to obtain a pure multivariate encryption scheme (such a scheme is proposed in Appendix B). This assumption is required to get formal results (Proposition 1, Lemma 1 and Proposition 4) but the function $\mathsf{Decrypt}$ does not require the factorization of n. This gives hope to remove this assumption: this basically consists of considering Schwartz-Zippel's lemma [Sch80] instead of Proposition 1 and adding randomness to the construction in order to maintain the truth of the formal results proved under the factoring assumption.

Notation. *We use standard Landau notations. Throughout this paper, we let* λ *denote the security parameter: all known attacks against the cryptographic scheme under scope should require* $2^{\Omega(\lambda)}$ *bit operations to mount. Let* $\kappa \geq 2$ *be an integer and let* $n = pq$ *be a randomly chosen RSA modulus. All the computations considered in this paper will be done in* $\mathbb{Z}_n$.

[1] It comes from the fact that $J_n(x) \mod p$ (resp. $J_n(x) \mod q$) is not a function of $x \mod p$ (resp. $x \mod q$)

– *Δ_κ is the set of permutations over $\{1, \ldots, \kappa\}$*

– *$\Sigma_\kappa = \{\sigma_1, \ldots, \sigma_\kappa\} \subset \Delta_\kappa$ defined*[2] *by $\sigma_i(j) = (i + j - 2 \bmod \kappa) + 1$.*

– *The cardinality of a set S will be denoted by $\#S$.*

– 'Choose at random $x \in X$' *will systematically mean that x is chosen according to uniform probability distribution over X.*

– 'An algorithm $\mathcal{A}$ outputs a polynomial p' *will systematically mean that $\mathcal{A}$ outputs a $\{+, -, \times\}$-circuit representing p.*

– *The inner product of two vectors $\boldsymbol{v}$ and $\boldsymbol{v}'$ is denoted by $\langle \boldsymbol{v}, \boldsymbol{v}' \rangle$*

– *The set of all square $t - by - t$ matrices over $\mathbb{Z}_n$ is denoted by $\mathbb{Z}_n^{t \times t}$.*

Remark 1. The number $M(m, d)$ of m-variate monomials of degree d is equal to $\begin{pmatrix} d + m - 1 \\ d \end{pmatrix}$. In particular, $M(2\kappa, \kappa) \approx 6^\kappa / \sqrt{\kappa}$.

2 Overview

In this section, we propose a high-level description of the main ideas of this paper. All the computations will be done in $\mathbb{Z}_n$, $n \geq 3$.

First encryption scheme. The secret key K contains 2κ randomly chosen secret vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{2\kappa}$ belonging to $\mathbb{Z}_n^{2\kappa}$.
Encrypting $x \in \mathbb{Z}_n$ simply consists of randomly choosing $\boldsymbol{c} \in \mathbb{Z}_n^{2\kappa}$ satisfying

$$\frac{\langle \boldsymbol{s}_1, \boldsymbol{c} \rangle}{\langle \boldsymbol{s}_2, \boldsymbol{c} \rangle} + \cdots + \frac{\langle \boldsymbol{s}_{2\kappa-1}, \boldsymbol{c} \rangle}{\langle \boldsymbol{s}_{2\kappa}, \boldsymbol{c} \rangle} = x \tag{1}$$

In other words, by considering the $2\kappa - by - 2\kappa$ matrix S whose i^{th} row is $\boldsymbol{s}_i$ (assuming S invertible)

$$\boldsymbol{c} = S^{-1} \begin{pmatrix} r_1 x_1 \\ r_1 \\ \cdots \\ r_\kappa x_\kappa \\ r_\kappa \end{pmatrix}$$

where $(x_i, r_i)_{i=1,\ldots,\kappa}$ is randomly chosen in $(\mathbb{Z}_n \times \mathbb{Z}_n^*)^\kappa$ s.t. $x_1 + \cdots + x_\kappa = x$.

Security analysis. By multiplying each side of (1) by $\Phi_0'(\boldsymbol{c}) = \prod_{i=1}^{\kappa} \langle \boldsymbol{s}_{2i}, \boldsymbol{c} \rangle$, we get a degree-$\kappa$ polynomial equation in the form

$$\Phi_0(\boldsymbol{c}) - x\Phi_0'(\boldsymbol{c}) = \Phi_x(\boldsymbol{c}) = \sum_{t_1 + \cdots + t_{2\kappa} = \kappa} \alpha_{t_1, \ldots, t_{2\kappa}} c_1^{t_1} \cdots c_{2\kappa}^{t_{2\kappa}} = 0$$

[2] $\sigma_i(1) = i; \sigma_i(2) = i + 1; \ldots; \sigma_i(\kappa) = i - 1$.

where the coefficients $\alpha_{t_1,\ldots,t_{2\kappa}}$ are evaluations of degree-κ polynomials over S, x. As $\Phi_x(\boldsymbol{c}) = 0$ if and only if $\boldsymbol{c}$ is an encryption of x, the knowledge of Φ_x is sufficient to break IND-CPA security. Moreover, by sampling sufficiently many encryptions of x, the monomials of Φ_x can be recovered by solving a linear system. However, by choosing $\kappa = \Theta(\lambda)$, the number of monomials is exponential (see Remark 1), making this attack fail.

Homomorphic properties. The *vector sum* is not an additive homomorphic operator. But, contrarily to what we may intuitively think, this scheme has some homomorphic capabilities coming from the following observation

$$\frac{\langle \boldsymbol{s}_1, \boldsymbol{c}\rangle\langle \boldsymbol{s}_2, \boldsymbol{c}'\rangle + \langle \boldsymbol{s}_2, \boldsymbol{c}\rangle\langle \boldsymbol{s}_1, \boldsymbol{c}'\rangle}{\langle \boldsymbol{s}_2, \boldsymbol{c}\rangle\langle \boldsymbol{s}_2, \boldsymbol{c}'\rangle} + \cdots + \frac{\langle \boldsymbol{s}_{2\kappa-1}, \boldsymbol{c}\rangle\langle \boldsymbol{s}_{2\kappa}, \boldsymbol{c}'\rangle + \langle \boldsymbol{s}_{2\kappa}, \boldsymbol{c}\rangle\langle \boldsymbol{s}_{2\kappa-1}, \boldsymbol{c}'\rangle}{\langle \boldsymbol{s}_{2\kappa}, \boldsymbol{c}\rangle\langle \boldsymbol{s}_{2\kappa}, \boldsymbol{c}'\rangle} = x + x'$$

where $\boldsymbol{c}$ and $\boldsymbol{c}'$ are encryptions of respectively x and x'. This will be used to develop an additive homomorphic operator.

Second encryption scheme. This second encryption scheme is essentially the same as the first one except that we consider an operator Add achieving homomorphic additions. Given two encryptions $\boldsymbol{c}$ and $\boldsymbol{c}'$ of x and x', $\mathsf{Add}(\boldsymbol{c}, \boldsymbol{c}')$ returns an encryption $\boldsymbol{c}''$ defined by

$$\boldsymbol{c}'' = S^{-1}\begin{pmatrix} r_1 r_1'(x_1 + x_1') \\ r_1 r_1' \\ \cdots \\ r_\kappa r_\kappa'(x_\kappa + x_\kappa') \\ r_\kappa r_\kappa' \end{pmatrix}$$

where $\boldsymbol{c} = S^{-1}(r_1 x_1, r_1, \ldots, r_\kappa x_\kappa, r_\kappa)$ and $\boldsymbol{c}' = S^{-1}(r_1' x_1', r_1', \ldots, r_\kappa' x_\kappa', r_\kappa')$.

Security analysis. Unfortunately, the adjunction of Add brings weaknesses. Indeed, we can mount what we will call an *attack by linearization.* For concreteness, the CPA attacker can efficiently build the vector $\widetilde{\boldsymbol{c}}$ defined by

$$\widetilde{\boldsymbol{c}} = S^{-1}\begin{pmatrix} r_1^{\phi(n)}\phi(n)x_1 \\ r_1^{\phi(n)} \\ \cdots \\ r_\kappa^{\phi(n)}\phi(n)x_\kappa \\ r_\kappa^{\phi(n)} \end{pmatrix} = \cdot S^{-1}\begin{pmatrix} \phi(n)x_1 \\ 1 \\ \cdots \\ \phi(n)x_\kappa \\ 1 \end{pmatrix}$$

by recursively applying Add over $\boldsymbol{c}$. We let see the reader how to use it to totally break our scheme[3]. To overcome this, the factoring assumption should

[3] By considering 2κ randomly chosen encryptions $\boldsymbol{c}_1, \ldots \boldsymbol{c}_{2\kappa}$ of arbitrarily chosen plaintexts $x_1, \ldots, x_{2\kappa}$, the vectors $\widetilde{\boldsymbol{c}}_1, \ldots \widetilde{\boldsymbol{c}}_{2\kappa}$ can be generated as explained above. For any $i = 1, \ldots, 2\kappa$, it is ensured that $\langle \boldsymbol{v}, \widetilde{\boldsymbol{c}}_i\rangle = \phi(n)x_i$, with $\boldsymbol{v} = \boldsymbol{s}_1 + \boldsymbol{s}_3 + \cdots + \boldsymbol{s}_{2\kappa-1}$. Hence, by solving this linear system (where the variables are the components of $\boldsymbol{v}$) $\boldsymbol{v}$ can be recovered. This is sufficient to break the IND-CPA security of our scheme. Indeed, given a challenge encryption $\boldsymbol{c}$, the encrypted value x can be recovered, i.e. $x = \langle \boldsymbol{v}, \widetilde{\boldsymbol{c}}\rangle / \phi(n)$.

be introduced by choosing n as a RSA modulus. In this paper, we will show how to efficiently implement Add and we will prove IND-CPA security in the generic ring model under the factoring assumption assuming $\kappa = \Theta(\lambda)$. This represents the main result of this paper.

Removing the factoring assumption? They are many other ways to define an additive homomorphic operator. For instance, randomness can be introduced in Add to get an operator $\mathsf{Add}^{\mathsf{rand}}$ by defining $\boldsymbol{c}'' = \mathsf{Add}^{\mathsf{rand}}(\boldsymbol{c}, \boldsymbol{c}')$ by

$$\boldsymbol{c}'' = S^{-1} \begin{pmatrix} \rho_1(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(1)} r'_{\sigma'(1)} (x_{\sigma(1)} + x'_{\sigma'(1)}) \\ \rho_1(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(1)} r'_{\sigma'(1)} \\ \cdots \\ \rho_\kappa(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(\kappa)} r'_{\sigma'(\kappa)} (x_{\sigma(\kappa)} + x'_{\sigma'(\kappa)}) \\ \rho_\kappa(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(\kappa)} r'_{\sigma'(\kappa)} \end{pmatrix}$$

where σ, σ' are randomly (and secretely) chosen permutations of $\{1, \ldots, \kappa\}$ and $\rho_1, \ldots, \rho_\kappa$ are randomly (and secretely) chosen (e.g. quadratic) polynomials. By doing this, the above attack does not work anymore and the factoring assumption could be hopefully removed. We let it as a perspective (an example of implementation is proposed in Appendix B).

Perspective of FHEs. By the same way, one can efficiently implement operators $\mathcal{O}$ computing $\boldsymbol{c}'' = \mathcal{O}(\boldsymbol{c}, \boldsymbol{c}')$ defined by

$$\boldsymbol{c}'' = S^{-1} \begin{pmatrix} \rho_1(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(1)} r'_{\sigma'(1)} x_{\sigma(1)} x'_{\sigma'(1)} \\ \rho_1(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(1)} r'_{\sigma'(1)} \\ \cdots \\ \rho_\kappa(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(\kappa)} r'_{\sigma'(\kappa)} x_{\sigma(\kappa)} x'_{\sigma'(\kappa)} \\ \rho_\kappa(\boldsymbol{c}, \boldsymbol{c}') r_{\sigma(\kappa)} r'_{\sigma'(\kappa)} \end{pmatrix}$$

Roughly speaking, $\boldsymbol{c}''$ *stores* κ products $x_i x_j$. By combining several such well-chosen operators (at least κ) and the additive homomorphic operator, one can build a multiplicative homomorphic operator (by using the equality $xx' = \sum_{ij} x_i x'_j$).

Discussion. The first encryption scheme can be straightforwardly turned into a new noise-free cryptographic problem. The search version of this problem would consist of recovering the secret matrix S given sufficiently many encryptions of 0 and the decisional version would consist of distinguishing between encryptions of 0 and randomly chosen vectors. We believe this problem hard for any $n \geq 3$ assuming $\kappa = \Theta(\lambda)$. In our opinion, this problem could be fruitful in cryptography and could merit to be independently studied. We briefly saw natural ways to build homomorphic operators. We think that many other relevant constructions can be achieved.

3 Some security results under the factoring assumption

Throughout this section, n denotes a randomly chosen RSA-modulus. Given a function $\phi : \mathbb{Z}_n^r \rightarrow \mathbb{Z}_n$, $z_\phi \stackrel{\mathsf{def}}{=} \#\{x \in \mathbb{Z}_n^r | \phi(x) = 0\}/n^r$. Classically a polynomial will be said *null* (or identically null) if each coefficient of its expanded representation is equal to 0.

3.1 Roots of polynomials

The following result proved in [AM09] establishes that it is difficult to output a polynomial ϕ such that z_ϕ is non-negligible without knowing the factorization of n. The security of RSA in the generic ring model can be quite straightforwardly derived from this result (see [AM09]).

Theorem 1. ***(Lemma 4 of [AM09]).*** *Assuming factoring is hard, there is no p.p.t-algorithm $\mathcal{A}$ which inputs n and which outputs*[4] *a $\{+, -, \times\}$-circuit representing a non-null polynomial $\phi \in \mathbb{Z}_n[X]$ such that z_ϕ is non-negligible.*

Thanks to this lemma, showing that two polynomials[5] are equal with non-negligible probability becomes an algebraic problem: it suffices to prove that they are identically equal. This lemma is a very powerful tool which is the heart of the security proofs proposed in this paper. We extend this result to the multivariate case.

Proposition 1. *Assuming factoring is hard, there is no p.p.t algorithm $\mathcal{A}$ which inputs n and which outputs*[4] *a $\{+, -, \times\}$-circuit representing a non-null polynomial $\phi \in \mathbb{Z}_n[X_1, \ldots, X_r]$ such that z_ϕ is non-negligible.*

Proof. See Appendix D.
□

3.2 Symmetry

Let $\kappa \geq 2$ and $t \geq 1$ be positive integers polynomials in λ. Recall that Δ_κ denotes the set of the permutations over $\{1, \ldots, \kappa\}$. Throughout this section, we will consider an arbitrary subset $\Sigma \subseteq \Delta_\kappa$. Let y_1, y_2 be randomly chosen in $\mathbb{Z}_n$. It is well-known that recovering[6] y_1 with non-negligible probability given only $S = y_1 + y_2$ or $P = y_1 y_2$ is difficult assuming the hardness of factoring. In this section, we propose to extend this. The following definition naturally extends the classical definition of symmetric polynomials.

[4] with non-negligible probability (the coin toss being the choice of n and the internal randomness of $\mathcal{A}$)

[5] built without knowing the factorization of n

[6] y_1, y_2 are the roots of the polynomial $y^2 - Sy + P$.

Definition 1. *Consider the tuples of indeterminate* $(Y_\ell = (X_{\ell 1}, \ldots, X_{\ell t}))_{\ell=1,\ldots,\kappa}$. *A polynomial* $\phi \in \mathbb{Z}_n[Y_1, \ldots, Y_\kappa]$ *is* Σ*-symmetric if for any permutation* $\sigma \in \Sigma$*,*

$$\phi(Y_1, \ldots, Y_\kappa) = \phi(Y_{\sigma(1)}, \ldots, Y_{\sigma(\kappa)})$$

Let $\mathcal{P}$ be an arbitrary p.p.t algorithm which inputs n and outputs m Σ-symmetric polynomials $s_1, \ldots, s_m$ and a non Σ-symmetric polynomial π. We show that evaluating π only given evaluations of $s_1, \ldots, s_m$ is difficult.

Lemma 1. *Let* n *be a randomly chosen RSA modulus and* $(s_1, \ldots, s_m, \pi) \leftarrow \mathcal{P}(n)$. *Assuming the hardness of factoring, there is no p.p.t algorithm which outputs* $\pi(y)$ *given only* $s_1(y), \ldots, s_m(y)$ *with non-negligible probability over the choice*[7] *of* $n, y \xleftarrow{\$} \mathbb{Z}_n^{\kappa t}$.

Proof. See Appendix E.
□

4 An additively homomorphic private-key encryption scheme

We first propose a private-key encryption scheme. The homomorphic operator will be developed later.

Definition 2. *Let* λ *be a security parameter. The functions* KeyGen*,* Encrypt*,* Decrypt *are defined as follows:*

- KeyGen(λ). *Let* η, κ *be positive integers indexed by* λ*, let* n *be an* η*-bit RSA modulus chosen at random. Choose at random an invertible matrix* $S \in \mathbb{Z}_n^{2\kappa \times 2\kappa}$ *and let* $T = S^{-1}$. *The* i^{th} *row of* S *is denoted by* $\boldsymbol{s}_i$ *and* $\mathcal{L}_i$ *denotes the linear function defined by* $\mathcal{L}_i(\boldsymbol{v}) = \langle \boldsymbol{s}_i, \boldsymbol{v} \rangle$. *Output*

$$K = \{S\} \ ; \ pp = \{n, \kappa\}$$

- Encrypt$(K, pp, x \in \mathbb{Z}_n)$. *Choose at random* $r_1, \ldots, r_\kappa$ *in* $\mathbb{Z}_n^*$ *and* $x_1, \ldots, x_\kappa$ *in* $\mathbb{Z}_n$ *s.t.* $x_1 + \cdots + x_\kappa = x$. *Output*

$$\boldsymbol{c} = T \begin{pmatrix} r_1 x_1 \\ r_1 \\ \cdots \\ r_\kappa x_\kappa \\ r_\kappa \end{pmatrix}$$

- Decrypt$(K, pp, \boldsymbol{c} \in \mathbb{Z}_n^{2\kappa})$. *Output* $x = \sum_{\ell=1}^{\kappa} \mathcal{L}_{2\ell-1}(\boldsymbol{c}) / \mathcal{L}_{2\ell}(\boldsymbol{c})$.

[7] y uniform over $\mathbb{Z}_n^{\kappa t}$

Throughout this paper, $pp = \{n, \kappa\}$ will be assumed to be public. The homomorphic operator(s), developed later, will be included in pp. Proving correctness is straightforward by using the relation $x = r_1 x_1/r_1 + \ldots + r_\kappa x_\kappa/r_\kappa$. The function $\mathsf{Decrypt}$ can be represented as the ratio of two degree-κ polynomials $\Phi_0, \Phi'_0 \in \mathbb{Z}_n[X_1, \cdots, X_{2\kappa}]$ defined by

$$\Phi_0 = \sum_{\ell=1}^{\kappa} \mathcal{L}_{2\ell-1} \prod_{\ell' \neq \ell} \mathcal{L}_{2\ell'} \ ; \ \Phi'_0 = \prod_{\ell=1}^{\kappa} \mathcal{L}_{2\ell} \tag{2}$$

i.e.

$$\mathsf{Decrypt}(K, pp, \boldsymbol{c}) = \Phi_0(\boldsymbol{c})/\Phi'_0(\boldsymbol{c})$$

At this step, our scheme is not homomorphic in the sense that the vector sum is not an homomorphic operator. Indeed, $\boldsymbol{c}$ and $a \cdot \boldsymbol{c}$ encrypt the same message for any $a \in \mathbb{Z}_n^*$.

4.1 Externalizing the generation of $\boldsymbol{n}$

To clearly understand the role of the factoring assumption in our security proof, it is important to notice that the factorization of n is not used in KeyGen. Consequently, the generation of n could be externalized[8] (for instance generated by an oracle). In other words, n could be a public input of KeyGen. This means that all the polynomials considered in our security analysis are built without using the factorization of n implying that they are equal to 0 with negligible probability provided they are not null (according to Proposition 1).

4.2 A basic attack

We present here the most natural attack consisting of solving a linear system. Let $\boldsymbol{c} \leftarrow \mathsf{Encrypt}(K, pp, 0)$ be an encryption of 0. By definition[9], it is ensured that $\Phi_0(\boldsymbol{c}) = 0$. By considering several encryptions $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t$ of 0, we get an equation system $\Phi_0(\boldsymbol{c}_1) = 0, \ldots, \Phi_0(\boldsymbol{c}_t) = 0$.

The expanded representation of Φ_0 could be thus recovered[10] by solving a linear system whose variables are its monomial coefficients. However, this attack fails provided $\kappa = \Theta(\lambda)$ because the expanded representation of Φ_0 is exponential-size in this case (see Remark 1). For instance, by choosing $\kappa = 13$, the attack consists of solving a linear system with approximatively $5 \cdot 10^9$ variables.

It should be noticed that the previous equation system can be seen as a nonlinear system whose variables are the coefficients of S. Proposition 4 will ensure that this system cannot be solved assuming the hardness of factoring.

[8] ensuring that its factorization was forgotten just after its generation

[9] Φ_0 defined in (2), satisfies $\Phi_0(\boldsymbol{c}) = \sum_{\ell=1}^{\kappa} r_\ell x_\ell \prod_{\ell' \neq \ell} r_{\ell'} = 0$

[10] within a multiplicative factor

4.3 The additive operator

Let $S \leftarrow \mathsf{KeyGen}(\lambda)$. In this section, we will consider the quadratic polynomials $\mathcal{L}_{ij} \in \mathbb{Z}_n[U_1, \ldots, U_{2\kappa}, V_1, \ldots, V_{2\kappa}]$ defined by $\mathcal{L}_{ij}(\boldsymbol{u}, \boldsymbol{v}) = \mathcal{L}_i(\boldsymbol{u})\mathcal{L}_j(\boldsymbol{v})$.

Definition 3. *$\mathsf{AddGen}(S)$ outputs the expanded representation of the polynomials $q_1, \ldots, q_{2\kappa}$ defined by*

$$\begin{pmatrix} q_1 \\ \cdots \\ q_{2\kappa} \end{pmatrix} = T \begin{pmatrix} \mathcal{L}_{12} + \mathcal{L}_{21} \\ \mathcal{L}_{22} \\ \cdots \\ \mathcal{L}_{2\kappa-1,2\kappa} + \mathcal{L}_{2\kappa,2\kappa-1} \\ \mathcal{L}_{2\kappa,2\kappa} \end{pmatrix}$$

As each quadratic polynomial q_i has $O(\kappa^2)$ monomials, the running time of AddGen is $O(\kappa^4)$ (2κ sums of 2κ quadratic polynomials). The operator $\mathsf{Add} \leftarrow \mathsf{AddGen}(S)$ consists of evaluating the polynomials $q_1, \ldots, q_{2\kappa}$, i.e. $\mathsf{Add}(\boldsymbol{u}, \boldsymbol{v}) = (q_1(\boldsymbol{u}, \boldsymbol{v}), \ldots, q_{2\kappa}(\boldsymbol{u}, \boldsymbol{v}))$, leading to a running time in $O(\kappa^3)$. See Appendix A for a toy implementation of Add.

Proposition 2. *$\mathsf{Add} \leftarrow \mathsf{AddGen}(S)$ is a valid additive homomorphic operator.*

Proof. Straightforward (see Fig. 1).
□

$$\boxed{\mathsf{Add}\left(T \begin{pmatrix} r_1 x_1 \\ r_1 \\ \cdots \\ r_\kappa x_\kappa \\ r_\kappa \end{pmatrix}, T \begin{pmatrix} r'_1 x'_1 \\ r'_1 \\ \cdots \\ r'_\kappa x'_\kappa \\ r'_\kappa \end{pmatrix} \right) = T \begin{pmatrix} r_1 r'_1 (x_1 + x'_1) \\ r_1 r'_1 \\ \cdots \\ r_\kappa r'_\kappa (x_\kappa + x'_\kappa) \\ r_\kappa r'_\kappa \end{pmatrix}}$$

Fig. 1. Description of the additive operator $\mathsf{Add} \leftarrow \mathsf{AddGen}(S)$ showing that $\mathsf{Decrypt}(K, pp, \mathsf{Add}(\boldsymbol{c}, \boldsymbol{c}')) = \mathsf{Decrypt}(K, pp, \boldsymbol{c}) + \mathsf{Decrypt}(K, pp, \boldsymbol{c}')$.

As seen in Section 2, the operator Add introduces weaknesses provided the factorization of n is known.

Proposition 3. *IND-CPA security $\Rightarrow$ hardness of factoring.*

4.4 Efficiency

$\mathsf{Encrypting}/\mathsf{Decrypting}/\mathsf{Add}$ requires respectively $O(\kappa^2/\kappa^2/\kappa^3)$ modular multiplications. A ciphertext is a 2κ-vector in $\mathbb{Z}_n$, implying that the ratio of ciphertext size to plaintext size is 2κ. In terms of storage, Add contains $4\kappa^3 + 6\kappa^2$ elements of $\mathbb{Z}_n$, which leads to a space complexity in $O(|n|\kappa^3)$.

By considering $\kappa = 13$ as done in Section 4.2, evaluating Add requires around 10500 modular multiplications *vs* only one for Paillier's cryptosystem. Efficiency could be improved by choosing n as a prime (large or not) in constructions not requiring the factoring assumption. We propose an example of such a construction in Appendix B.

4.5 Discussion

The private-key encryption scheme is very simple. Many cryptographic constructions based on this scheme can be imagined by adding auxiliary information, e.g. the operator Add. For these reasons, we think that the security of this scheme can be seen as a new cryptographic problem and its security can be studied independently of related constructions.

The classic way (see [Rot11]) to transform a private-key cryptosystem into a public-key cryptosystem consists of publicizing encryptions $\boldsymbol{c}_i$ of known values x_i and using the homomorphic operators to encrypt x. Let $\mathsf{Encrypt1}$ denote this new encryption function. Assuming the IND-CPA security of the private-key cryptosystem, it suffices that $\mathsf{Encrypt1}(pk, x)$ and $\mathsf{Encrypt}(K, pp, x)$ are computationally indistinguishable to ensure the IND-CPA security of the public-key cryptosystem.

5 Security analysis

Notation. *Let* $Y = ((X_{i\ell}, R_{i\ell})_{i=0,\ldots,t}, (S_{2\ell-1,i}, S_{2\ell,i})_{i=1,\ldots,2\kappa})_{\ell=1,\ldots,\kappa}$ *be a tuple of indeterminate used throughout this section. Typically, a polynomial* $\alpha \in \mathbb{Z}_n[Y]$ *will be evaluated over* θ_n, θ_n *containing the randomness used to build the knowledge of the CPA attacker (see Definition 4) and* $\alpha(\theta_n)$ *being a value known by the CPA attacker.*

Breaking IND-CPA security consists of recovering a p.p.t. algorithm $\mathcal{A}$ distinguishing encryptions of 0 from ones of 1, i.e. satisfying

$$|\mathsf{Pr}(\mathcal{A}(\mathsf{Encrypt}(K, pp, 1)) = 0) - \mathsf{Pr}(\mathcal{A}(\mathsf{Encrypt}(K, pp, 0)) = 0)| > \varepsilon(\lambda) \qquad (3)$$

where $\varepsilon(\lambda)$ is a non-negligible quantity. Throughout our security analysis, it will be assumed that

$$\kappa = \Theta(\lambda)$$

5.1 Knowledge of the CPA attacker.

For technical reasons, we propose a slight modification in Definitions 2, 3 by setting $T = \det^2 S \cdot S^{-1}$ (instead of $T = S^{-1}$): each coefficient of T can be thus expressed as a polynomial defined over S keeping true some symmetry properties encapsulated in Lemma 2. It is straightforward to show that the decrypting function and the operator Add remain correct.

There are classically two sources of randomness *behind* the knowledge of the CPA attacker. The first source of randomness is the internal randomness of KeyGen, i.e. the choice of $K = \{S\}$. The second source of randomness comes from the encryption oracle. After receiving the challenge encryption $\boldsymbol{c}_0 \leftarrow \mathsf{Encrypt}(K, pp, x_0)$, the CPA attacker requests the encryption oracle to get encryptions $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t$ of chosen plaintexts $x_1, \ldots, x_t \in \mathbb{Z}_n$. Without loss of generality, we will here assume that the encryptions are **random** meaning that the encryption oracle randomly chooses[11] plaintexts $x_1, \ldots, x_t$ itself and returns these values and their encryptions $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t$ (drawn according to $\mathsf{Encrypt}$). This assumption can be done because the CPA attacker can use the operator Add, after receiving $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t, x_1, \ldots, x_t$, to get encryptions of chosen plaintexts statistically indistinguishable from encryptions output by $\mathsf{Encrypt}$. Clearly, it suffices to consider $t = O(\kappa)$ to ensure this. All the randomness can be encapsulated in the vector θ_n defined as follows.

Definition 4. *Let $S \leftarrow \mathsf{KeyGen}(\lambda)$, let $(x_{i1}, r_{i1}, \ldots, x_{i\kappa}, r_{i\kappa})$ be the values (randomly) chosen by the encryption oracle to produce*[12] *$\boldsymbol{c}_i$. For any $\ell \in \{1, \ldots, \kappa\}$, the random vector $\theta_\ell \in \mathbb{Z}_n^{4\kappa+2(t+1)}$ is defined by*

$$\theta_\ell = ((x_{i\ell}, r_{i\ell})_{i=0,\ldots,t}, (s_{2\ell-1,i}, s_{2\ell,i})_{i=1,\ldots,2\kappa})$$

The random vector $(\theta_1, \ldots, \theta_\kappa)$ is denoted by θ_n if $x_0 = x_{01} + \cdots + x_{0\kappa}$ is uniform over $\mathbb{Z}_n$ and $\theta_n^{[x]}$ if $x_0 = x$.

It should be noticed that θ_n is drawn according to a probability statistically indistinguishable from the uniform distribution over $\mathbb{Z}_n^{\kappa\gamma}$. The knowledge of the CPA attacker can be represented as a vector $\boldsymbol{\alpha} \in \mathbb{Z}_n^{\gamma'}$, with $\gamma' = O(\kappa^3)$ provided $t = \Theta(\kappa)$.

Definition 5. *The CPA attacker's knowledge $(\boldsymbol{c}_0, \ldots, \boldsymbol{c}_t, x_1, \ldots, x_t, \mathsf{Add})$ can be represented by a vector $\boldsymbol{\alpha} \in \mathbb{Z}_n^{\gamma'}$, the i^{th} component of $\boldsymbol{\alpha}$ being the evaluation of a polynomial*[16] *$\alpha_i \in \mathbb{Z}_n[Y]$ over θ_n, i.e. $\boldsymbol{\alpha} = (\alpha_1(\theta_n), \ldots, \alpha_{\gamma'}(\theta_n)) \stackrel{def}{=} \alpha(\theta_n)$.*

The polynomials α_i are implicitly described in previous sections. Nevertheless, we do not need to precisely define them. We will only exploit their symmetry properties. For instance, Add is not impacted by switching the two first rows of S with the two last ones. The following result generalizes it.

Lemma 2. *Each polynomial α_i is Δ_κ-symmetric (see Definition 1).*

Proof. See Appendix F.
□

[11] according the the uniform distribution over $\mathbb{Z}_n$.
[12] $\boldsymbol{c}_i = T(r_{i1}x_{i1}, r_{i1}, \ldots, r_{i\kappa}x_{i\kappa}, r_{i\kappa})$.

5.2 A fundamental result based on symmetry

By exploiting intrinsic symmetry properties of our scheme, one can show that S cannot be recovered. Worse, non Δ_κ-symmetric polynomials cannot be evaluated over the secret matrix S.

Proposition 4. *Let*[16] $\pi \in \mathbb{Z}_n[Y]$ *be a non* Δ_κ*-symmetric polynomial chosen by the CPA attacker* $\mathcal{A}$*. Assuming the hardness of factoring,* $\mathcal{A}$ *cannot recover* $\pi(\theta_n)$ *with non-negligible probability over the choice of* θ_n, n*.*

Proof. A direct consequence of Lemma 1 and Lemma 2.
□

Corollary 1. *Assume the hardness of factoring.*

1. *The secret key* S *cannot be recovered.*
2. *Any product of strictly less than* κ *coefficients of* S *cannot be recovered.*
3. *The polynomials* $\mathcal{L}_{i_1} \times \cdots \times \mathcal{L}_{i_t}$ *cannot be recovered*[13] *provided* $t < \kappa$*.*

This result is not sufficient to ensure that $\Phi_0 = \sum_{\ell=1}^{\kappa} \mathcal{L}_{2\ell-1} \prod_{\ell' \neq \ell} \mathcal{L}_{2\ell'}$ cannot be recovered. Indeed, each monomial coefficient of Φ_0 is Δ_κ-symmetric (and thus could be recovered). However, the expanded representation of Φ_0 (or its multiples) is exponential-size provided $\kappa = \Theta(\lambda)$ and thus cannot be recovered.

By construction, Φ_0 (or its multiples) could nevertheless be efficiently represented with the linear functions $\mathcal{L}_i$ (or O(1)-products of these linear functions). However, these compact semi-factored representations do not deal with Δ_κ-symmetric quantities and they cannot be recovered according to Proposition 4. However, maybe other efficient representations of Φ_0 can exist only dealing with Δ_κ-symmetric values. We will show that it is not the case in the generic ring model (see Proposition 8) which is sufficient to prove generic IND-CPA security (see Proposition 7).

5.3 Attacks by linearization

Proposition 4 intuitively justifies that our security analysis can be restricted to a natural class of attacks, called *attacks by linearization*, generalizing the attacks described in Sections 2 and 4.2. For concreteness, the CPA attacker $\mathcal{A}$ can generate new vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r$ by recursively applying the homomorphic operator Add on the challenge encryption $\boldsymbol{c}_0$ and $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t$ in the hope that there exists a *small* polynomial φ s.t. $\Phi(\boldsymbol{c}_0) = \varphi(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r)$ distinguishes between encryptions of 0 and encryptions of 1. For instance, $\boldsymbol{v}_1 = \mathsf{Add}(\boldsymbol{c}_0, \boldsymbol{c}_0)$, $\boldsymbol{v}_2 = \mathsf{Add}(\boldsymbol{v}_1, \boldsymbol{c}_0)$, $\boldsymbol{v}_3 = \mathsf{Add}(\boldsymbol{v}_2, \boldsymbol{c}_1)$, etc. The procedure (chosen by the attacker) which outputs $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r)$ is denoted by GenVec, i.e. $\Phi(\boldsymbol{c}_0) = \varphi \circ \mathsf{GenVec}(\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_t)$. If the expanded representation of φ is small enough then the CPA attacker could recover it by solving a linear system.

[13] and thus cannot be evaluated

Proposition 5. *Assuming the hardness of factoring, the CPA attacker cannot find*[14] *a procedure* GenVec *and a polynomial-size polynomial*[15] $\varphi \in \mathbb{Z}_n[X_1, \ldots, X_{2\kappa r}]$ *s.t.* $\varphi \circ \mathsf{GenVec}$ *satisfies*

$$|Pr_{c_0 \leftarrow \mathsf{Encrypt}(pk,pp,1)}(\varphi \circ \mathsf{GenVec}(\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_t) = 0) \\ - Pr_{c_0 \leftarrow \mathsf{Encrypt}(pk,pp,0)}(\varphi \circ \mathsf{GenVec}(\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_t) = 0)| > \varepsilon(\lambda)$$

with non-negligible probability over the choice of $(\boldsymbol{c}_i \leftarrow \mathsf{Encrypt}(pk, pp, x_i))_{i=1,\ldots,t}$

Proof. See Appendix G.
□

5.4 Generic IND-CPA security

Roughly speaking, a Generic Ring Algorithm (GRA) defined over a ring $\mathcal{R}$ (here $\mathcal{R} = \mathbb{Z}_n$) is an algorithm where only arithmetic operations $+, -, \times, /$ and equality tests are allowed (see [AM09]). In the special case of $\mathcal{R} = \mathbb{Z}_n$ where n is a randomly chosen RSA modulus, equality tests are not needed. This is implicitly shown in [AM09] as a straightforward consequence of Theorem 1. Indeed, this result ensures that two polynomials are either identically equal or equal with negligible probability. We say that our scheme is secure in the generic ring model if the CPA cannot find any distinguishing rational function.

Definition 6. *Our encryption scheme is generically IND-CPA secure if the CPA attacker cannot recover a* $\{+, -, \times, /\}$*-circuit representing a (rational) function* ϕ *satisfying*

$$\left| Pr\left(\phi \circ \alpha(\theta_n^{[1]}) = 0\right) - Pr\left(\phi \circ \alpha(\theta_n^{[0]}) = 0\right) \right| > \varepsilon(\lambda) \tag{4}$$

where $\varepsilon(\lambda)$ *is a non-negligible quantity.*

This definition can be restricted to polynomials.

Proposition 6. *Our encryption scheme is generically IND-CPA secure if the CPA attacker cannot recover a (polynomial-size)* $\{+, -, \times\}$*-circuit representing a polynomial* ϕ *satisfying (4).*

Proof. See Appendix H.1
□

To prove generic security, we will prove that the CPA attacker cannot output a non-null polynomial ϕ such that $\phi \circ \alpha(\theta_n^{[x]}) = 0$ with non-negligible probability. Without loss of generality, we will focus on the case $x = 0$. In this case, the polynomial ϕ_0 defined as follows plays a central role in our analysis.

---

[14] with non-negligible probability

[15] polynomial-size expanded representation. Note that degree-κ polynomials have an exponential number of monomials (see Remark 1) provided $\kappa = \Theta(\lambda)$.

Definition 7. *Let us consider the polynomials*[16] $L_t(Y,V) = \sum_{k=1}^{2\kappa} S_{t,k} \cdot V_k$ *with* $V = (V_1, \ldots, V_{2\kappa})$*. The polynomial* $\phi_0 \in \mathbb{Z}_n[Y,V]$ *is defined by*

$$\phi_0 = \sum_{\ell=1}^{\kappa} L_{2\ell-1} \prod_{\ell' \neq \ell} L_{2\ell'}$$

By construction, the polynomial ϕ_0 satisfies $\phi_0(\theta_n, \boldsymbol{v}) = \Phi_0(\boldsymbol{v})$. The following proposition states that our scheme is generically IND-CPA secure if the CPA attacker cannot represent any non-null multiple of ϕ_0 from its knowledge. To simplify notation, we redefine α by $\alpha(\theta_n, \boldsymbol{v}) = (\alpha_1(\theta_n), \ldots, \alpha_{\gamma'}(\theta_n), \boldsymbol{v})$.

Proposition 7. *Assuming the hardness of factoring, our scheme is generically IND-CPA secure if and only if the CPA attacker cannot output*[17] *a (polynomial-size)* $\{+,-,\times\}$*-circuit representing a polynomial* ϕ *s.t.* $\phi \circ \alpha$ *is a non-null multiple of* ϕ_0 *(see Definition 7).*

Proof. See appendix H.2.
□

Consequently, generic IND-CPA security can be reduced to an algebraic problem. Indeed, it suffices to prove the non-existence of polynomials ϕ satisfying requirements of Proposition 7. The proof is based on the Δ_κ-symmetry of CPA attacker's knowledge (see Lemma 2).

Proposition 8. *Our scheme is generically IND-CPA secure assuming the hardness of factoring.*

Proof. See Appendix H.3.
□

This result holds as long as Lemma 2 holds. It means in particular that IND-CPA security is ensured even if other evaluations of Δ_κ-symmetric polynomials are given to the CPA attacker.

6 Perspectives

A first motivating perspective would consist of removing the factoring assumption required to prove formal results (Theorem 1, Lemma 1 and Proposition 4). This assumption defeats the whole "post-quantum" purpose of multivariate cryptography [Pat96]. While decrypting does not require the factorization of n, this assumption allows us to prove some formal impossibility results. Randomness might be introduced in order to get a pure multivariate encryption scheme. In our opinion, the additional randomness introduced to develop the multiplicative operator (in the following of this section) could be sufficient to achieve this (such randomness should also be introduced in Add).

[16] Recall that $Y = ((X_{i\ell}, R_{i\ell})_{i=0,\ldots,t}, (S_{2\ell-1,i}, S_{2\ell,i})_{i=1,\ldots,2\kappa})_{\ell=1,\ldots,\kappa}$.
[17] with non-negligible probability over the choice of n

6.1 A naive/toy construction of Mult

We here consider the case $\kappa = 2$ where S is a 4×4 matrix. Let us consider the two following quadratic operators $\mathcal{O}_1, \mathcal{O}_2$ defined by (see Section 4.3 for notation) :

$$\mathcal{O}_1 = T\begin{pmatrix}\mathcal{L}_{11}\\ \mathcal{L}_{22}\\ \mathcal{L}_{33}\\ \mathcal{L}_{44}\end{pmatrix} ; \mathcal{O}_2 = T\begin{pmatrix}\mathcal{L}_{13}\\ \mathcal{L}_{24}\\ \mathcal{L}_{31}\\ \mathcal{L}_{42}\end{pmatrix}$$

Given two encryptions $\boldsymbol{c}, \boldsymbol{c}'$ of x, x', we have

$$\mathcal{O}_1(\boldsymbol{c}, \boldsymbol{c}') = T\begin{pmatrix} r_1 r'_1 x_1 x'_1\\ r_1 r'_1\\ r_2 r'_2 x_2 x'_2\\ r_2 r'_2\end{pmatrix} ; \mathcal{O}_2(\boldsymbol{c}, \boldsymbol{c}') = T\begin{pmatrix} r_1 r'_2 x_1 x'_2\\ r_1 r'_2\\ r_2 r'_1 x_2 x'_1\\ r_2 r'_1\end{pmatrix}$$

implying that $\boldsymbol{c}'' = \mathsf{Mult}(\boldsymbol{c}, \boldsymbol{c}') \stackrel{\mathsf{def}}{=} \mathsf{Add}(\mathcal{O}_1(\boldsymbol{c}, \boldsymbol{c}'), \mathcal{O}_2(\boldsymbol{c}, \boldsymbol{c}'))$ is a valid encryption of xx'. Indeed,

$$\boldsymbol{c}'' = T\begin{pmatrix} r_1^2 r'_1 r'_2 (x_1 x'_1 + x_1 x'_2)\\ r_1^2 r'_1 r'_2\\ r_2^2 r'_1 r'_2 (x_2 x'_1 + x_2 x'_2)\\ r_2^2 r'_1 r'_2\end{pmatrix}$$

and $\mathsf{Decrypt}(K, pp, \boldsymbol{c}'') = (x'_1 + x'_2)x_1 + (x'_1 + x'_2)x_2 = (x'_1 + x'_2)(x_1 + x_2) = xx'$.

Roughly speaking, the $\kappa^2 = 4$ products $x_i x'_j$ are stored in two intermediate vectors output by $\mathcal{O}_1, \mathcal{O}_2$. While there are many others ways to define these operators, let us assume that their description is public[18] (or guessed by the CPA attacker). This choice of $\mathcal{O}_1, \mathcal{O}_2$ leads to an attack by linearization more efficient than the basic attack presented in Section 4.2.

Example of attack by linearization. Assume that $\boldsymbol{c}'$ is an encryption of 0, i.e. $x' = x'_1 + x'_2 = 0$. In this case[19],

$$\mathsf{Mult}(\boldsymbol{c}, \boldsymbol{c}') \sim T\begin{pmatrix}0\\ r_1^2\\ 0\\ r_2^2\end{pmatrix}$$

It follows that a linear combination of $\mathcal{L}_1, \mathcal{L}_3$ can be recovered by solving a small linear system[20] allowing the CPA attacker to distinguish the case $x' = 0$ from the case $x' \neq 0$. In order to remove such weaknesses, we will introduce randomness in our construction, i.e. the coefficients τ_{ijk} and the polynomials ρ_{ijk}.

[18] while the operators are public, their description could be not divulged.
[19] $\sim$ meaning "equal within a multiplicative constant"
[20] smaller than the one involved in the basic attack.

6.2 Overview

A multiplicative operator Mult should be developed to get an FHE. Let $\boldsymbol{c}, \boldsymbol{c}'$ be two encryptions of x, x'. The operator Mult developed in this section will output an encryption $\boldsymbol{c}'' = \mathsf{Mult}(\boldsymbol{c}, \boldsymbol{c}')$ satisfying

$$\boldsymbol{c}'' = T \begin{pmatrix} R_1(\boldsymbol{c}, \boldsymbol{c}') \cdot \sum_{ij} \tau_{ij1} x_i x'_j \\ R_1(\boldsymbol{c}, \boldsymbol{c}') \\ \cdots \\ R_\kappa(\boldsymbol{c}, \boldsymbol{c}') \cdot \sum_{ij} \tau_{ij\kappa} x_i x'_j \\ R_\kappa(\boldsymbol{c}, \boldsymbol{c}') \end{pmatrix}$$

where τ_{ijk} are randomly chosen over $\mathbb{Z}_n$ s.t. $\sum_{k=1}^{\kappa} \tau_{ijk} = 1$ for any $(i,j) \in \{1, \ldots, \kappa\}^2$ and $R_1, \ldots, R_\kappa$ are randomly chosen polynomials. Clearly,

$$\mathsf{Decrypt}(K, pp, \boldsymbol{c}'') = \sum_k \sum_{ij} \tau_{ijk} x_i x'_j = \sum_{ij} x_i x'_j = xx'$$

Unfortunately, unlike Add, this operator Mult cannot be efficiently represented with Δ_κ-symmetric values. We propose to represent it by using weaker symmetry properties.

The implementation of Mult is less straightforward than the one of Add. It cannot be achieved using only one quadratic operator. Indeed, it exploits the equality $xx' = \sum_{i=1}^{\kappa} \sum_{j=1}^{\kappa} x_i x'_j$ and several operators are necessary to *store* all the products $x_i x'_j$ in some intermediate vectors. The price to pay is to degrade symmetry properties. Nevertheless, we propose a construction partially keeping them.

6.3 Our proposal

Notation. *Let $I_\kappa = \{1, \ldots, \kappa\}$ and let Γ^κ be the set of quadratic homogeneous polynomials $\rho \in \mathbb{Z}_n[X_1, \ldots, X_{2\kappa}, Y_1, \ldots, Y_{2\kappa}]$ s.t. $\rho(X, Y) = \sum_{i,j} a_{ij} X_i Y_j$.*

Given two permutations $\sigma, \sigma' \in \Delta_\kappa$, a family of polynomials $\rho \in \Gamma^\kappa$ and a vector $\tau \in \mathbb{Z}_n^\kappa$, the function $\mathsf{OGen}(S, \sigma, \sigma', \rho, \tau)$ outputs[21] the degree-4 operator $\mathcal{O}$ defined by

$$\mathcal{O} = T \begin{pmatrix} \tau_1 & \rho_1 \mathcal{L}_{2\sigma(1)-1, 2\sigma'(1)-1} \\ & \rho_1 \mathcal{L}_{2\sigma(1), 2\sigma'(1)} \\ & \cdots \\ \tau_\kappa & \rho_\kappa \mathcal{L}_{2\sigma(\kappa)-1, 2\sigma'(\kappa)-1} \\ & \rho_\kappa \mathcal{L}_{2\sigma(\kappa), 2\sigma'(\kappa)} \end{pmatrix}$$

By construction,

$$\mathsf{Decrypt}(sk, \mathcal{O}(\boldsymbol{c}, \boldsymbol{c}')) = \tau_1 x_{\sigma(1)} x_{\sigma'(1)'} + \cdots + \tau_\kappa x_{\sigma(\kappa)} x_{\sigma'(\kappa)'}$$

[21] the expanded representation of the 2κ degree-4 polynomials $q_1, \ldots, q_{2\kappa}$ satisfying $(q_1(\boldsymbol{u}, \boldsymbol{v}), \ldots, q_{2\kappa}(\boldsymbol{u}, \boldsymbol{v})) = \mathcal{O}(\boldsymbol{u}, \boldsymbol{v})$.

We note that $\mathsf{Decrypt}(sk, \mathcal{O}(\boldsymbol{c}, \boldsymbol{c}'))$ does not depend on the polynomials ρ_i. These polynomials will be chosen at random in Mult. Roughly speaking, the vector $\mathcal{O}(\boldsymbol{c}, \boldsymbol{c}')$ stores κ (additive shares of) products $x_i x_j$. By considering several such operators (at least κ), all the products can be stored. It then suffices to homomorphically add these vectors (by using the operator Add) to get an encryption of xx'. This is detailed below.

Mult. Let $\tau = (\tau_{ijk})_{(i,j,k)\in I_\kappa^3}$ be randomly chosen such that $\sum_{k=1}^{\kappa} \tau_{ijk} = 1$ for any $(i,j) \in I_\kappa^2$. To build the operator Mult, it suffices to invoke κ^2 times the function OGen in order to generate and publicize

$$\mathcal{O}_{ij} \leftarrow \mathsf{OGen}\left(S, \sigma_i, \sigma_j, \rho_{ij}, (\tau_{\sigma_i(k),\sigma_j(k),k})_{k=1,\ldots,\kappa}\right)$$

for any $(i,j) \in I_\kappa^2$ where ρ_{ij} is randomly chosen over Γ^κ and $\sigma_i, \sigma_j \in$ [22] Σ_κ. To homomorphically multiply $\boldsymbol{c}$ and $\boldsymbol{c}'$, it suffices to homomorphically add the vectors $\mathcal{O}_{ij}(\boldsymbol{c}, \boldsymbol{c}')$, i.e.

$$\mathsf{Mult}(\boldsymbol{c}, \boldsymbol{c}') \stackrel{\mathsf{def}}{=} \bigoplus_{(i,j)\in I_\kappa^2} \mathcal{O}_{ij}(\boldsymbol{c}, \boldsymbol{c}')$$

where $\oplus$ refers to the operator Add, i.e. $\boldsymbol{u} \oplus \boldsymbol{v} = \mathsf{Add}(\boldsymbol{u}, \boldsymbol{v})$. As evaluating $\mathcal{O}_{ij}(\boldsymbol{c}, \boldsymbol{c}')$ can be done in $O(\kappa^5)$, the running time of Mult is $O(\kappa^7)$.

Example. Description of the operators $\mathcal{O}_{11}$, $\mathcal{O}_{12}$, $\mathcal{O}_{12}$, $\mathcal{O}_{22}$ and Mult in the case $\kappa = 2$.

$$\mathcal{O}_{11}(\boldsymbol{c}, \boldsymbol{c}') = T \begin{pmatrix} \rho_{111}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_1' \tau_{111} x_1 x_1' \\ \rho_{111}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_1' \\ \rho_{112}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_2' \tau_{222} x_2 x_2' \\ \rho_{112}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_2' \end{pmatrix}; \quad \mathcal{O}_{12}(\boldsymbol{c}, \boldsymbol{c}') = T \begin{pmatrix} \rho_{121}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_2' \tau_{121} x_1 x_2' \\ \rho_{121}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_2' \\ \rho_{122}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_1' \tau_{212} x_2 x_1' \\ \rho_{122}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_1' \end{pmatrix}$$

$$\mathcal{O}_{21}(\boldsymbol{c}, \boldsymbol{c}') = T \begin{pmatrix} \rho_{211}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_2' \tau_{211} x_2 x_1' \\ \rho_{211}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_2' \\ \rho_{212}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_1' \tau_{122} x_1 x_2' \\ \rho_{212}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_1' \end{pmatrix}; \quad \mathcal{O}_{22}(\boldsymbol{c}, \boldsymbol{c}') = T \begin{pmatrix} \rho_{221}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_2' \tau_{221} x_2 x_2' \\ \rho_{221}(\boldsymbol{c}, \boldsymbol{c}') r_2 r_2' \\ \rho_{222}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_1' \tau_{112} x_1 x_1' \\ \rho_{222}(\boldsymbol{c}, \boldsymbol{c}') r_1 r_1' \end{pmatrix}$$

$$\mathsf{Mult}(\boldsymbol{c}, \boldsymbol{c}') \sim T \begin{pmatrix} \prod_{(i,j)\in\{1,2\}^2} \rho_{ij1}(\boldsymbol{c}, \boldsymbol{c}') \sum_{(i,j)\in\{1,2\}^2} \tau_{ij1} x_i x_j' \\ \prod_{(i,j)\in\{1,2\}^2} \rho_{ij1}(\boldsymbol{c}, \boldsymbol{c}') \\ \prod_{(i,j)\in\{1,2\}^2} \rho_{ij2}(\boldsymbol{c}, \boldsymbol{c}') \sum_{(i,j)\in\{1,2\}^2} \tau_{ij2} x_i x_j' \\ \prod_{(i,j)\in\{1,2\}^2} \rho_{ij2}(\boldsymbol{c}, \boldsymbol{c}') \end{pmatrix}$$

---

[22] Recall that $\sigma_i \in \Sigma_\kappa$ refers to the permutation over $\{1, \ldots, \kappa\}$ defined by $\sigma_i(1) = i; \sigma_i(2) = i+1; \ldots; \sigma_i(\kappa) = i-1$.

6.4 Security analysis

Randomness θ_n (see Definition 4) can be easily adapted in order to integrate the polynomials ρ_{ijk} and the values τ_{ijk} used in our construction. Each value known by the CPA attacker can be still written as the evaluation of a polynomial α_i (see Definition 5) over θ_n. In this context, Proposition 7 remains true. Unfortunately, Proposition 8 cannot be naturally extended because its proof is based on that the polynomials α_i are Δ_κ-symmetric. Even if Lemma 2 is not true anymore, the polynomials α_i keep symmetry properties: they are just Σ_κ-symmetric instead of being Δ_κ-symmetric. Proposition 4 can be easily adapted.

Proposition 9. *Let π be a non Σ_κ-symmetric polynomial chosen by the CPA attacker $\mathcal{A}$. Assuming the hardness of factoring, $\mathcal{A}$ cannot recover $\pi(\theta_n)$ with non-negligible probability over the choice of θ_n, n.*

Proof. In order to take into account (symmetric) constraints over the coefficients τ_{ijk}, a slight extension of Lemma 1, i.e. Lemma 6, should be used to prove this result.
□

It follows that Corollary 1 still holds. However, the proof of Proposition 8 intrinsically exploits Δ_κ-symmetry properties and cannot be easily adapted. While we are convinced that the introduction of the polynomials ρ_{ijk} and the coefficients τ_{ijk} protect our scheme against attacks by linearization, we did not manage to formally prove it.

Assume nevertheless that the multiplicative operator Mult can be replaced by an oracle $\mathcal{O}$ in the security analysis. In this case, the proof of Proposition 5 can be easily adapted to show the non-existence of efficient attacks by linearization.

Mult can be replaced by an oracle $\mathcal{O}$? We propose two (informal) reasons/modifications suggesting this.

- The operators $\mathcal{O}_{ij}$ play a symmetric role and there is no reason to publicize the permutations σ_i, σ_j involved in these operators. We can speculate on the fact that the CPA attacker cannot recover them or equivalently that it cannot distinguish between $\mathcal{O}_{ij}$ and $\mathcal{O}_{i'j'}$.
- The operators $\mathcal{O}_{ij}$ output vectors relevant under the secret key S. However, nothing justifies it and one can imagine that $\mathcal{O}_{ij}$ output vectors relevant under randomly chosen keys S_{ij}. It suffices then to generate new operators Add (adapted to these new keys) in order to (homomorphically) add these vectors. Roughly speaking, the operators $\mathcal{O}_{ij}$ and the (new) operators Add involved in Mult become chained making non-specified uses irrelevant[23].

[23] This also could lead to significative improvements by replacing each degree-4 operators by two quadratic operators.

References

[AM09] Divesh Aggarwal and Ueli M. Maurer. Breaking RSA generically is equivalent to factoring. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, pages 36–53, 2009.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. Cryptology ePrint Archive, Report 2011/344, 2011. http://eprint.iacr.org/.

[CGGI18] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: fast fully homomorphic encryption over the torus. *IACR Cryptology ePrint Archive*, 2018:421, 2018.

[CL15] Guilhem Castagnos and Fabien Laguillaumie. Linearly homomorphic encryption from $$\mathsf {DDH}$$. In *Topics in Cryptology - CT-RSA 2015, The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings*, pages 487–505, 2015.

[CNT12] 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.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 617–640, 2015.

[Elg85] T. Elgamal. A public key cryptosystem and a signature sheme based on discrete logarithms. In *IEEE transactions on Information Theory*, pages 31:469–472, 1985.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[GHS12a] Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In *EUROCRYPT*, pages 465–482, 2012.

[GHS12b] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the aes circuit. In *CRYPTO*, pages 850–867, 2012.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 75–92, 2013.

[JS09] Tibor Jager and Jörg Schwenk. On the analysis of cryptographic assumptions in the generic ring model. In *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, pages 399–416, 2009.

[KH12] Aviad Kipnis and Eliphaz Hibshoosh. Efficient methods for practical fully homomorphic symmetric-key encrypton, randomization and verification. Cryptology ePrint Archive, Report 2012/637, 2012. http://eprint.iacr.org/.

[LNV11] Kristin Lauter, Michael Naehrig, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? *IACR Cryptology ePrint Archive*, 2011:405, 2011.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.

[Pat96] Jacques Patarin. Hidden fields equations (hfe) and isomorphisms of polynomials (ip): Two new families of asymmetric algorithms. In *EUROCRYPT*, pages 33–48, 1996.

[Rot11] Ron Rothblum. *Homomorphic Encryption: From Private-Key to Public-Key*, pages 219–234. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, October 1980.

[SS10] Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In *ASIACRYPT*, pages 377–394, 2010.

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010.

[XBY12] Liangliang Xiao, Osbert Bastani, and I-Ling Yen. An efficient homomorphic encryption protocol for multi-user systems. *IACR Cryptology ePrint Archive*, 2012:193, 2012.

A Implementation of Add in the case $\kappa = 1$

In this section, we provide an example of the implementation of the homomorphic scheme for $\kappa = 1$. Let $S = [s_{ij}] \in \mathbb{Z}_n^{2\times 2}$ and $\Delta = s_{11}s_{22} - s_{12}s_{21}$.

The polynomial $\mathsf{Add} = (q_1, q_2) \leftarrow \mathsf{AddGen}(S)$ are defined by

$$\begin{aligned}\Delta \cdot q_1(\boldsymbol{u}, \boldsymbol{v}) =& (2s_{22}s_{11}s_{21} - s_{12}s_{21}^2)u_1v_1 \\ &+ s_{22}^2 s_{11}(u_1v_2 + u_2v_1) \\ &+ s_{12}s_{22}^2 u_2v_2\end{aligned}$$

$$\begin{aligned}\Delta \cdot q_2(\boldsymbol{u}, \boldsymbol{v}) =& s_{11}s_{21}^2 u_1v_1 \\ &- s_{21}^2 s_{12}(u_1v_2 + u_2v_1) \\ &+ (s_{11}s_{22}^2 - 2s_{21}s_{12}s_{22})u_2v_2\end{aligned}$$

B Removing the factoring assumption?

We propose to implement the *randomized* operator $\mathsf{Add}^{\mathsf{rand}}$ considered in Section 2 (with $\sigma = \sigma' = \mathsf{Id}$). This operator can be implemented with degree-4 polynomials (provided the polynomials ρ_i are quadratic). To improve efficiency, we propose to split it into two quadratic operators Add' and Rand, i.e.

$$\mathsf{Add}^{\mathsf{rand}}(\boldsymbol{c}, \boldsymbol{c}') = \mathsf{Rand}(\mathsf{Add}'(\boldsymbol{c}, \boldsymbol{c}'))$$

- Add' exactly follows Add except that $\boldsymbol{c}'' = \mathsf{Add}'(\boldsymbol{c}, \boldsymbol{c}')$ is not relevant under S but relevant under a randomly chosen S'.
- Rand *randomizes* $\boldsymbol{c}''$ with polynomials ρ_i.

Let $S' \in \mathbb{Z}_n^{2\kappa\times 2\kappa}$ be a randomly chosen invertible matrix, $T' = S'^{-1}$ its inverse and $\mathcal{L}'_i$ the linear application defined by $\mathcal{L}'_i(\boldsymbol{u}) = \langle \boldsymbol{s}'_i, \boldsymbol{u}\rangle$ where $\boldsymbol{s}'_i$ is the i^{th} row of S'.

Add'. It suffices now to define Add' as Add except that T is replaced by T' in Definition 3. In other words,

$$\mathsf{Add}'(\boldsymbol{c}, \boldsymbol{c'}) = T' \cdot S \cdot \mathsf{Add}(\boldsymbol{c}, \boldsymbol{c'})$$

Rand. Let $\rho_1, \ldots, \rho_\kappa$ be randomly chosen degree-1 polynomials.

$$\mathsf{Rand}(\boldsymbol{u}) = T \begin{pmatrix} \rho_1(\boldsymbol{u})\mathcal{L}'_1(\boldsymbol{u}) \\ \rho_1(\boldsymbol{u})\mathcal{L}'_2(\boldsymbol{u}) \\ \cdots \\ \rho_\kappa(\boldsymbol{u})\mathcal{L}'_{2\kappa-1}(\boldsymbol{u}) \\ \rho_\kappa(\boldsymbol{u})\mathcal{L}'_{2\kappa}(\boldsymbol{u}) \end{pmatrix}$$

It is straightforward to see that this new operator $\mathsf{Add}^{\mathsf{rand}}$ is correct

Security Analysis. As symmetry properties are preserved, all the results proved previously still hold under the factoring assumption. Let us now assume that n is a prime (instead of a RSA modulus). Note first that n should be a large prime, i.e. $n \approx 2^\lambda$, to avoid that $\rho_i(\boldsymbol{u}) = 0$ with non-negligible probability. Clearly, the attack by linearization exhibited in Section 2 is not relevant anymore. However, as the factorization of n is known, nonlinear univariate equations can be solved. Hence, our construction becomes potentially vulnerable to attacks based on Groëbner bases. We carry out some experiments on *SageMath platform* using variable elimination algorithms. It appears that computational-time required by these attacks is prohibitive even for very small values of κ, e.g $\kappa = 2$. We did not exhibit any attack working faster than the basic attack (see Section 4.2). Obviously further investigations should be done. In our opinion this is a nice challenge whose formulation is relatively simple.

C Some algebraic lemmas

Lemma 3. *Let t, r be positive integers such that $r \leq t$ and $a \in \mathbb{Z}_n$. Let $\phi \in \mathbb{Z}_n[X_1, \dots, X_t]$ be a polynomial and let φ be the $(t-1)$-variate polynomial defined by $\varphi(X_1, \dots, X_{r-1}, X_{r+1}, \dots, X_t) = \phi(X_1, \dots, X_{r-1}, a-(X_1+\dots+X_{r-1}), X_{r+1}, \dots, X_t)$. The polynomial φ is null if and only if ϕ can be factored by $(X_1 + \dots + X_r - a)$.*

Proof. Without loss of generality, one proves the result for $r = t$. We can identify $\mathbb{Z}_n[X_1, \dots, X_t]$ to $R[X_t]$ with $R = \mathbb{Z}_n[X_1, \dots, X_{t-1}]$. Let $\phi \in R[X]$. To state our result, it suffices to notice that ϕ can be factored by $X - (a - X_1 + \dots + X_{t-1})$ if and only if $\phi(a - (X_1 + \dots + X_{t-1})) = 0$.
$\square$

Lemma 4. *There do not exist any polynomial $q \in \mathbb{Z}_n[X_1, \dots, X_t]$ and symmetric polynomials*[24] *$\pi_1, \dots, \pi_t \in \mathbb{Z}_n[X_1, \dots, X_\kappa]$ satisfying $\deg \pi_i < \kappa$ and $q(\pi_1, \dots, \pi_t) = X_1 \cdots X_\kappa$.*

Proof. Let $\pi_1, \dots, \pi_t \in \mathbb{Z}_n[X_1, \dots, X_\kappa]$ be arbitrary symmetric polynomials s.t. $\deg \pi_i < \kappa$. Let us consider the κ symmetric polynomials $\sigma_k = \sum_{1 \leq i_1 < \dots < i_k < \kappa} X_{i_1} \cdots X_{i_k}$ and an arbitrary symmetric polynomial $\phi \in \mathbb{Z}_n[X_1, \dots, X_\kappa]$. The fundamental theorem of symmetric polynomials says that there exists a unique polynomial φ satisfying $\phi = \varphi(\sigma_1, \dots, \sigma_\kappa)$. Thus, as $\deg \pi_i < \kappa$, $\pi_1, \dots, \pi_t$ can be written as polynomials φ_i defined over $\sigma_1, \dots, \sigma_{\kappa-1}$ but σ_κ cannot. Thus, there is no polynomial $q \in \mathbb{Z}_n[X_1, \dots, X_t]$ s.t. $q(\pi_1, \dots, \pi_t) = \sigma_\kappa = X_1 \cdots X_\kappa$.
$\square$

Lemma 5. *Let $\varphi \in \mathbb{Z}_n[X_1, \dots, X_{2\kappa}]$ be a polynomial. Assume that the polynomial $\phi \in \mathbb{Z}_n[X_1, \dots, X_{2\kappa}]$ defined by*

$$\phi(X_1, \dots, X_{2\kappa}) = \varphi(X_1 X_2, X_2, \dots, X_{2\kappa-1} X_{2\kappa}, X_{2\kappa})$$

[24] $\pi_1, \dots, \pi_t \in \mathbb{S}_1$.

can be factored by $X_1 + X_3 + \cdots + X_{2\kappa-1}$. It is ensured that φ can be factored by $\psi(X_1, \ldots, X_{2\kappa}) = \sum_{\ell=1,\ldots,\kappa} X_{2\ell-1} \prod_{\ell' \neq \ell} X_{2\ell'}$.

Proof. (Sketch.) Clearly,

$$\varphi(X_1X_2, X_2, \ldots, X_{2\kappa-1}X_{2\kappa}, X_{2\kappa}) \neq \varphi'(X_1X_2, X_2, \ldots, X_{2\kappa-1}X_{2\kappa}, X_{2\kappa})$$

provided $\varphi \neq \varphi'$.

By construction, each monomial $X_1^{e_1}X_2^{e_2} \cdots X_{2\kappa-1}^{e_{2\kappa-1}}X_{2\kappa}^{e_{2\kappa}}$ of ϕ satisfies $e_{2\ell-1} < e_{2\ell}$. It follows that if ϕ is a multiple of $X_1 + X_3 + \ldots + X_{2\kappa-1}$ then ϕ can be factored by $X_2X_4 \cdots X_{2\kappa}$ and thus by $X_2X_4 \cdots X_{2\kappa}(X_1 + X_3 + \ldots + X_{2\kappa-1})$, i.e. there exists φ' such that

$$\begin{aligned} &\phi(X_1, \ldots, X_{2\kappa}) \\ =&X_2X_4 \cdots X_{2\kappa}(X_1 + X_3 + \ldots + X_{2\kappa-1})\varphi'(X_1X_2, X_2, \ldots, X_{2\kappa-1}X_{2\kappa}, X_{2\kappa}) \end{aligned}$$

We conclude by noticing that

$$\psi(X_1X_2, X_2 \ldots, X_{2\kappa-1}X_{2\kappa}, X_{2\kappa}) = X_2X_4 \cdots X_{2\kappa}(X_1 + X_3 + \ldots + X_{2\kappa-1})$$

implying that $\varphi = \psi \cdot \varphi'$.

□

D Proof of Proposition 1

This result can be shown by induction over r. By Lemma 1, the result is true for $r = 1$. Let us assume the result true for any $r < t$ and let us show it for $r = t$. We can identify $\mathbb{Z}_n[X_1, \ldots, X_t]$ to $R[X_t]$ with $R = \mathbb{Z}_n[X_1, \ldots, X_{t-1}]$. Let ϕ be a a non-null polynomial $\phi \in \mathbb{Z}_n[X_1, \ldots, X_t]$ output by a p.p.t. algorithm $\mathcal{A}$, i.e. $\phi \leftarrow \mathcal{A}(n)$. ϕ can be identified by a non-null polynomial $\phi' \in R[X_1]$. Thus, by fixing $X_2, \ldots, X_t$ to randomly chosen values $x_2, \ldots, x_t \in \mathbb{Z}_n$, the polynomial $\phi_{x_2,\ldots,x_t}$ defined by $\phi_{x_2,\ldots,x_t}(x_1) = \phi(x_1, \ldots, x_t)$ is not (identically) null with overwhelming probability over the choice of $n, x_2, \ldots, x_t$ according to the induction hypothesis. Moreover, provided $\phi_{x_2,\ldots,x_t}$ is not null, $\phi_{x_2,\ldots,x_t}(x_1) = 0$ with negligible probability other choice of n, x_1 according to the induction hypothesis. This proves $\phi(x_1, \ldots, x_t) = 0$ with negligible over the choice of $n, x_1, \ldots, x_t$.

□

E Proof of Lemma 1

E.1 The proof

Let D be the uniform probability distribution of over $\mathbb{Z}_n^{\kappa t}$ The proof consists of building a polynomial factoring algorithm $\mathcal{A}$ by using a solver $\mathcal{B}$ of our problem as subroutine[25]. Let us consider the following polynomial-time algorithm $\mathcal{A}$:

[25] $\mathcal{B}$ is assumed to solve our problem if it outputs $\pi(y)$ with non-negligible probability

Input: $n = pq$

$(s_1, \ldots, s_m, \pi) \leftarrow \mathcal{P}(n)$

Repeat

1. *Let* $y = (y_1, \ldots, y_\kappa) \stackrel{\$}{\leftarrow} D$
2. *Compute* $\overline{s}_j = s_j(y)$ *for all* $j = 1, \ldots, m$.
3. *Compute* $\Pi = \pi(y)$
4. *Apply* $\mathcal{B}$ *on the inputs* $\overline{s}_1, \ldots, \overline{s}_m$, *i.e.* $\Pi_{\mathcal{B}} \leftarrow \mathcal{B}(\overline{s}_1, \ldots, \overline{s}_m)$

until $\gcd(\Pi - \Pi_{\mathcal{B}}, n) \neq 1$

output $\gcd(\Pi - \Pi_{\mathcal{B}}, n)$

By construction, this algorithm is correct. Let us show that it terminates in polynomial-time. First, each step of $\mathcal{A}$ can be computed in polynomial-time implying that $\mathcal{A}$ is polynomial if the expectation of the number of steps of $\mathcal{A}$ is polynomial (or equivalently, if the probability to get $\gcd(\Pi - \Pi_{\mathcal{B}}, n) \neq 1$ is not negligible).

As π is not Σ-symmetric, there exists $\sigma^* \in \Sigma$ s.t. $\pi - \pi_{\sigma^*}$ is not null, where π_{σ^*} is the polynomial defined by $\pi_{\sigma^*}(y) = \pi(y_{\sigma^*(1)}, \ldots, y_{\sigma^*(\kappa)})$. Thus, according to Proposition 1, $\pi(y) \neq \pi_{\sigma^*}(y)$ with overwhelming probability. It follows that $\pi(y) \not\equiv \pi_{\sigma^*}(y) \mod p$ or $\pi(y) \not\equiv \pi_{\sigma^*}(y) \mod q$ with overwhelming probability. Without loss of generality, we assume that

$$\pi(y) \not\equiv \pi_{\sigma^*}(y) \mod q \tag{5}$$

with overwhelming probability. Let us consider the function $h : (\mathbb{Z}_n^t)^\kappa \to (\mathbb{Z}_n^t)^\kappa$ such that $(y'_1, \ldots, y'_\kappa) = h(y_1, \ldots, y_\kappa)$ is defined by

- $y'_{\ell i} \equiv y_{\ell i} \mod p$ for any $(\ell, i) \in \{1, \ldots, \kappa\} \times \{1, \ldots, t\}$
- $y'_{\ell i} \equiv y_{\sigma^*(\ell), i} \mod q$ for any $(\ell, i) \in \{1, \ldots, \kappa\} \times \{1, \ldots, t\}$.

Obviously, $y' = (y'_1, \ldots, y'_\kappa)$ are y have the same probability over D, i.e.

$$\mathsf{Pr}_D(y) = \mathsf{Pr}_D(y')$$

Let $\Pi' = \pi(y')$. As the functions s_j are Σ-symmetric polynomials, we get $s_j(y') = s_j(y)$ for all $j = 1, \ldots, m$. It follows that

$$\mathsf{Pr}_D(\Pi_{\mathcal{B}} = \Pi) = \mathsf{Pr}_D(\Pi_{\mathcal{B}} = \Pi')$$

As $\mathcal{B}$ is assumed to solve our problem, $\mathsf{Pr}_D(\Pi_{\mathcal{B}} = \Pi)$ is non-negligible implying that $\mathsf{Pr}_D(\Pi_{\mathcal{B}} = \Pi')$ is non-negligible.

By construction $\Pi \equiv \Pi' \mod p$. Since $\Pi' \equiv \pi_{\sigma^*}(y) \mod q$, Equation (5) implies that $\Pi \not\equiv \Pi' \mod q$ with overwhelming probability. It follows that $p = \gcd(n, \Pi - \Pi')$ with overwhelming probability. Consequently, $\mathcal{A}$ terminates (when $\Pi_{\mathcal{B}} = \Pi'$) in polynomial-time.

□

E.2 Extension

We now propose to extend this result when y is drawn under symmetric constraints. Let assume that $\mathcal{A}_S(n)$ outputs:

- Σ-symmetric polynomials $s_1, \ldots, s_m \in \mathbb{Z}_n[((X_{ij}, Z_{ij})_{j=1,\ldots,t})_{i=1,\ldots,\kappa}]$
- polynomials $p_1, \ldots, p_\gamma \in \mathbb{Z}_n[((Z_{ij})_{j=1,\ldots,t})_{i=1,\ldots,\kappa}]$
- a non Σ-symmetric polynomial $\pi \in \mathbb{Z}_n[((X_{ij})_{j=1,\ldots,t})_{i=1,\ldots,\kappa}]$.

We consider the probability distribution $D^{p_1,\ldots,p_\gamma,\Sigma}$ uniform over the set (assumed to be not empty)

$$\{((x_1, z_1), \ldots, (x_\kappa, z_\kappa)) \in \mathbb{Z}_n^{(t+r)\kappa} | p_i(z_{\sigma(1)}, \ldots, z_{\sigma(\kappa)}) = 0 \\ \text{for any } (i, \sigma) \in \{1, \ldots, \gamma\} \times \Sigma\}$$

We will assume that $D^{p_1,\ldots,p_\gamma,\Sigma}$ is sampleable meaning there exists a p.p.t. algorithm $\mathcal{D}$ s.t. $\mathcal{D}(n)$ outputs a vector drawn according to a probability distribution statistically close to $D^{p_1,\ldots,p_\gamma,\Sigma}$.

Lemma 6. *Let $(s_1, \ldots, s_m, p_1, \ldots, p_\gamma, \pi) \leftarrow \mathcal{A}_S(n)$. Assuming the hardness of factoring, there is no p.p.t algorithm which outputs $\pi(x_1, \ldots, x_\kappa)$ given only*[26] *$s_1(y), \ldots, s_m(y)$ with non-negligible probability over the choice of n, $y = ((x_1, z_1), \ldots, (x_\kappa, z_\kappa)) \leftarrow D^{p_1,\ldots,p_\gamma,\Sigma}$.*

Proof. Exactly follows the proof of lemma 1.
□

F Proof of Lemma 2

Recall that we set $T = \det^2 S \cdot S^{-1}$ in order to ensure that value known by the CPA attacker can be written as the evaluation of a polynomial over θ. It remains to prove that these polynomials are Δ_κ-symmetric. First, it should be noticed that[27] $\det^2 S$ can be written as a Δ_κ-symmetric polynomial defined over $\boldsymbol{s} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{2\kappa})$ and thus $\theta_n = (\theta_1, \ldots, \theta_\kappa)$. The values $x_i = x_{i1} + \cdots + x_{i\kappa}$ are also evaluations of Δ_κ-polynomials.

By construction, each component of $\boldsymbol{c}_i$ is the evaluation over θ_n of a Δ_κ-symmetric polynomial. Indeed, $\boldsymbol{c}_i$ is the unique vector satisfying the following system

$$\text{for any } \ell = 1, \ldots, \kappa \quad \begin{cases} \langle s_{2\ell-1}, \boldsymbol{c}_i \rangle = (\det S)^2 \cdot r_{i\ell} x_{i\ell} \\ \langle s_{2\ell}, \boldsymbol{c}_i \rangle = (\det S)^2 \cdot r_{i\ell} \end{cases}$$

stable by permutating the tuples $(\theta_1, \ldots, \theta_\kappa)$.

[26] and an efficient representation of $\pi, s_1, \ldots, s_m$.
[27] but not $\det S$.

Let $(q_1, \ldots, q_{2\kappa}) \leftarrow \mathsf{AddGen}(S)$. The coefficient of $u_i v_j$ in $q_k(u,v)$ is denoted by a_{kij}. By construction, the vector $a_{ij} = (a_{1ij}, \ldots, a_{2\kappa,ij})$ is the unique solution of the following linear system (the variables being a_{kij})

$$\text{for any } \ell = 1, \ldots, \kappa \quad \begin{cases} \langle s_{2\ell-1}, a_{ij}\rangle = (\det S)^2 \cdot s_{2\ell-1,i}s_{2\ell,j} + s_{2\ell,i}s_{2\ell-1,j} \\ \langle s_{2\ell}, a_{ij}\rangle = (\det S)^2 \cdot s_{2\ell,i}s_{2\ell,j} \end{cases}$$

stable by permutating the tuples $(\theta_1, \ldots, \theta_\kappa)$. It follows that a_{kij} is the evaluation over θ_n of a Δ_κ-symmetric polynomial.
□

G Proof of Proposition 5

Lemma 7. *Let $\phi \in \mathbb{Z}_n[X_1, \ldots, X_\kappa, Y_1, \ldots, Y_\kappa]$ be a non-null polynomial such that each monomial $X_1^{e_1} \cdots X_\kappa^{e_\kappa} Y_1^{e'_1} \cdots Y_\kappa^{e'_\kappa}$ satisfies*

- $\exists i \in \{1, \ldots, \kappa\}$, $e_i = e'_i = 0$
- $\forall i \in \{1, \ldots, \kappa\}$, $e_i = 0 \Rightarrow e'_i = 0$

For any $\alpha \in \mathbb{Z}_n$, the polynomial $\phi_\alpha = \phi(X_1, \ldots, X_\kappa, Y_1, \ldots, Y_{\kappa-1}, \alpha - Y_1 - \ldots - Y_{\kappa-1})$ is not null.

Proof. Let $\phi = \sum_{i=1}^{\rho} a_i M_i$ where $M_i = X_1^{e_{i1}} \cdots X_\kappa^{e_{i,\kappa}} Y_1^{e'_{i1}} \cdots Y_\kappa^{e'_{i,\kappa}}$ and $a_i \in \mathbb{Z}_n^*$, let $m = \max_i e'_{i,\kappa}$.

If $m = 0$ then the result is trivially true. Thus, one can assume that $m > 0$. We have $\phi_\alpha = \sum_{i=0}^{\rho} a_i (\alpha - Y_1 - \ldots - Y_{\kappa-1})^{e'_{i,\kappa}} M'_i$ where $M'_i = X_1^{e_{i,1}} \cdots X_\kappa^{e_{i,\kappa}} Y_1^{e'_{i,1}} \cdots Y_{\kappa-1}^{e'_{i,\kappa-1}}$.

Given an arbitrary monomial $M = X_1^{e_1} \cdots X_\kappa^{e_\kappa} Y_1^{e'_1} \cdots Y_\kappa^{e'_\kappa}$, the set $\{j \in \{1, \ldots, \kappa\} | e_j \neq 0\}$ is denoted by $E(M)$. Let i_0 s.t. $e'_{i_0,\kappa} = m$. As $\exists j \in \{1, \ldots, \kappa\}$ s.t. $e_{ij} = e_{i'j} = 0$, one can assume that $1 \notin E(M'_{i_0})$. Let us show that the monomial $Y_1^m M'_{i_0}$ belongs to ϕ_α (implying that ϕ_α is not null). To achieve this, it suffices to show that this monomial does not belong to any polynomial $(\alpha - Y_1 - \ldots - Y_{\kappa-1})^{e'_{i,\kappa}} M'_i$ with $i \neq i_0$.

Suppose that there exists $i_1 \neq i_0$ s.t. $Y_1^m M'_{i_0}$ belongs to $(\alpha - Y_1 - \ldots - Y_{\kappa-1})^{e'_{i_1,\kappa}} M'_{i_1}$. Clearly, $1 \notin E(M'_{i_0})$ implies that $1 \notin E(M'_{i_1})$ and $e'_{i_1,\kappa} \geq m$ (because the constraint $e_i = 0 \Rightarrow e'_i = 0$ implies that the exponent of Y_1 in M'_{i_1} is equal to 0). By definition of m, it follows that $e'_{i_1,\kappa} = m$ implying that $M'_{i_0} \neq M'_{i_1}$ (because $M_{i_0} = M_{i_1}$ otherwise). Thus, $Y_1^m M'_{i_0}$ does not belong to $(\alpha - Y_1 - \ldots - Y_{\kappa-1})^{e'_{i_1,\kappa}=m} M'_{i_1}$. This concludes the proof.
□

Let us assume that the CPA attacker can recover a procedure GenVec and a polynomial φ of $\mathbb{Z}_n[X_1, \ldots, X_{2\kappa r}]$ such that $\deg \varphi < \kappa$ satisfying the requirements of the proposition. Let $\boldsymbol{c}_1^*, \ldots, \boldsymbol{c}_t^*$ be encryptions such that

$$\begin{aligned} |\mathsf{Pr}_{c_0 \leftarrow \mathsf{Encrypt}(pk,pp,1)}&(\varphi \circ \mathsf{GenVec}(\boldsymbol{c}_0, \boldsymbol{c}_1^*, \ldots, \boldsymbol{c}_t^*) = 0) \\ &- \mathsf{Pr}_{c_0 \leftarrow \mathsf{Encrypt}(pk,pp,0)}(\varphi \circ \mathsf{GenVec}(\boldsymbol{c}_0, \boldsymbol{c}_1^*, \ldots, \boldsymbol{c}_t^*) = 0)| > \varepsilon(\lambda) \end{aligned}$$

It follows that the polynomial $\overline{\varphi \circ \mathsf{GenVec}} \in \mathbb{Z}_n[R_1, \ldots, R_\kappa, X_1, \ldots, X_\kappa]$ defined by

$$\overline{\varphi \circ \mathsf{GenVec}}(R_1, \ldots, R_\kappa, X_1, \ldots, X_\kappa) = \varphi \circ \mathsf{GenVec}(Y, \boldsymbol{c}_1^*, \ldots, \boldsymbol{c}_t^*)$$

where[28] $Y = T(R_1X_1, R_1, \ldots, R_\kappa X_\kappa, R_\kappa)$ is not null. By construction of GenVec, each vector $\boldsymbol{v}$ output by $\varphi \circ \mathsf{GenVec}(Y, \boldsymbol{c}_1^*, \ldots, \boldsymbol{c}_t^*)$ is in the form

$$\boldsymbol{v} = T\left(p_1(R_1X_1, X_1), p_1'(R_1X_1, R_1), \ldots, p_\kappa(R_\kappa X_\kappa, R_\kappa), p_\kappa'(R_\kappa X_\kappa, R_\kappa)\right)$$

where p_i, p_i' are polynomials.

Consequently, as $\deg \varphi < \kappa$, each monomial $R_1^{e_1} \cdots R_\kappa^{e_\kappa} X_1^{e_1'} \cdots X_\kappa^{e_\kappa'}$ of $\overline{\varphi \circ \mathsf{GenVec}}$ satisfies

- $\exists i \in \{1, \ldots, \kappa\}$ s.t. $e_i = e_i' = 0$
- $\forall i \in \{1, \ldots, \kappa\}$, $e_i = 0 \Rightarrow e_i' = 0$.

Let $x \in \mathbb{Z}_n$ be arbitrarily chosen. By fixing $X_1 + \cdots + X_\kappa = x$, we consider the polynomial $\overline{\varphi \circ \mathsf{GenVec}_x} \in \mathbb{Z}_n[R_1, \ldots, R_\kappa, X_1, \ldots, X_{\kappa-1}]$ equal to the polynomial $\overline{\varphi \circ \mathsf{GenVec}}(R_1, \ldots, R_\kappa, X_1, \ldots, X_{\kappa-1}, x - X_{\kappa-1} - \ldots - X_1)$. By Lemma 7, this polynomial is not null. Hence, according to Proposition 1,

$$\overline{\varphi \circ \mathsf{GenVec}_x}(r_1, \ldots, r_\kappa, x_1, \ldots, x_{\kappa-1}) = 0$$

with negligible probability over the choice of $r_1, \ldots, r_\kappa \in \mathbb{Z}_n^*$ and $x_1, \ldots, x_{\kappa-1} \in \mathbb{Z}_n$ assuming factoring is hard. Thus, for any $x \in \mathbb{Z}_n$,

$$\mathsf{Pr}_{\boldsymbol{c}_0 \leftarrow \mathsf{Encrypt}(pk, pp, x)}(\varphi \circ \mathsf{GenVec}(\boldsymbol{c}_0, \boldsymbol{c}_1^*, \ldots, \boldsymbol{c}_t^*) = 0)$$

is negligible leading to a contradiction implying that $\deg \varphi \geq \kappa$.

□

H Proofs of Section 5.4

H.1 Proof of Proposition 6

Given $\mathcal{C}$ be a polynomial-size $\{+, -, \times, /\}$-circuit, we denote by $\phi_{\mathcal{C}}$ the (rational) function computing by $\mathcal{C}$. In [AM09], by induction on the gates of $\mathcal{C}$, it is shown that there exists a p.p.t. algorithm $\mathcal{A}$ such that $\mathcal{A}(\mathcal{C})$ outputs two polynomial-size $\{+, -, \times\}$-circuits $\mathcal{C}', \mathcal{C}''$ satisfying $\phi_{\mathcal{C}} = \phi_{\mathcal{C}'}/\phi_{\mathcal{C}''}$. Let us assume that ϕ_C satisfies (4). According to Proposition 1, if $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[x]})$ is not null then it is equal to 0 with negligible probability. Firstly, $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[0]})$ and $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[1]})$ cannot be both null because $\phi_{\mathcal{C}}$ satisfies (4). If $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[1]})$ is null but not $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[0]})$ (or the converse) then $\phi_{\mathcal{C}''}$ satisfies (4). Finally, if $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[1]})$ and $\phi_{\mathcal{C}''} \circ \alpha(\theta_n^{[0]})$ are both not null then $\phi_{\mathcal{C}'}$ satisfies (4). This proves that the CPA attacker can recover a polynomial satisfying (4).

□

[28] $T = \det^2 S \cdot S^{-1}$

H.2 Proof of Proposition 7

Recall that $\theta_n = ((x_{i\ell}, r_{i\ell})_{i=1,\dots,r}, s_{2\ell-1}, s_{2\ell})_{\ell=1,\dots,\kappa}$ is drawn according to a probability distribution statistically close to the uniform one over $\mathbb{Z}_n^\gamma$, with $\gamma = 4\kappa^2 + 2(t+1)\kappa$.

Consider the tuples of indeterminate $S = (S_{ij})_{(i,j)\in\{1,\dots,\kappa\}^2}$, $V = (V_i)_{i\in\{1,\dots,2\kappa\}}$, $Y = ((X_{i\ell}, R_{i\ell})_{i=0,\dots,t}, (S_{2\ell-1,i}, S_{2\ell,i})_{i=1,\dots,2\kappa})_{\ell=1,\dots,\kappa}$ and $Z = (Y, V)$. By construction, Z has $\gamma + 2\kappa$ components.

Let $T = [t_{ij}] = (\det^2 S)S^{-1}$. The degree-$(4\kappa - 1)$ polynomial computing t_{ij} is (abusively) denoted by t_{ij}, i.e. $t_{ij}(S) = t_{ij}$. We also consider the degree-4κ polynomial Δ defined by $\Delta(Z) = \det^2(S)$ and the polynomials I_j defined by $I_j(Z_1, \dots, Z_{\gamma+2\kappa}) = Z_j$.

Let assume that the CPA attacker can recover a non-null polynomial $\Psi \in \mathbb{Z}_n[X_1, \dots, X_{\gamma'+2\kappa}]$ such that $\Psi(\alpha_1(\theta_n), \dots, \alpha_{\gamma'}(\theta_n), \boldsymbol{c}) = 0$ with non-negligible probability over $\theta_n, \boldsymbol{c} \leftarrow \mathsf{Encrypt}(K, pp, 0)$. Let $\psi, \delta_1, \dots, \delta_{2\kappa}, \varepsilon_1, \dots, \varepsilon_{2\kappa}, \nu_1, \dots, \nu_\kappa$ be polynomials defined by

- $\psi(Z) = \Psi(\alpha_1(Y), \dots, \alpha_{\gamma'}(Y), V)$
- $\delta_\ell(Z) = \sum_{j=1}^{2\kappa} S_{\ell j} V_j$
- $\varepsilon_\ell(Z) = \sum_{j=1}^{2\kappa} t_{\ell j}(S) V_j$
- $\nu_\ell(Y, X_1, \dots, X_\kappa, R_1, \dots, R_\kappa) = R_\ell X_\ell$

We consider the polynomial tuples :
- $\delta = (I_1, \dots, I_\gamma, \delta_1, \dots, \delta_{2\kappa})$
- $\varepsilon = (I_1, \dots, I_\gamma, \varepsilon_1, \dots, \varepsilon_{2\kappa})$
- $\nu = (I_1, \dots, I_\gamma, \Delta \cdot \nu_1, \Delta \cdot I_{\gamma+\kappa+1} \dots, \Delta \cdot \nu_\kappa, \Delta \cdot I_{\gamma+2\kappa})$

By construction,

$$\varepsilon \circ \delta \stackrel{\mathsf{def}}{=} (\varepsilon_1(\delta), \dots, \varepsilon_{\gamma+2\kappa}(\delta)) = (I_1, \dots, I_\gamma, \Delta \cdot I_{\gamma+1}, \dots \Delta \cdot I_{\gamma+2\kappa})$$

It follows that the CPA attacker can recover a polynomial[29] ψ' and $t \in \mathbb{N}$ satisfying

$$\Delta^t \cdot \psi = \psi' \circ \varepsilon \circ \delta$$

By construction, given $\boldsymbol{c} = T(r_1 x_1, r_1, \dots, r_\kappa x_\kappa, r_\kappa) \leftarrow \mathsf{Encrypt}(K, pp, 0)$, it is ensured that

$$\begin{aligned}&\nu(\theta_n, x_1, \dots, x_\kappa, r_1, \dots, r_\kappa)\\ &\qquad = (\theta_n, \Delta(\theta_n, \boldsymbol{c}) \cdot (r_1 x_1, r_1, \dots, r_\kappa x_\kappa, r_\kappa)) = \delta(\theta_n, \boldsymbol{c})\end{aligned}$$

As $\Delta^t \cdot \psi(\theta_n, \boldsymbol{c}) = \psi' \circ \varepsilon \circ \delta(\theta_n, \boldsymbol{c}) = \psi' \circ \varepsilon \circ \nu(\theta_n, x_1, \dots, x_\kappa, r_1, \dots, r_\kappa)$,

$$\psi' \circ \varepsilon \circ \nu(\theta_n, x_1, \dots, x_\kappa, r_1, \dots, r_\kappa) = 0$$

with non-negligible probability provided $x_\kappa = -x_1 - \dots - x_{\kappa-1}$. Thus, according to Proposition 1

$$\psi' \circ \varepsilon \circ \nu(Y, X_1, \dots, X_{\kappa-1}, -(X_1 + \ldots + X_{\kappa-1}), R_1, \dots, R_\kappa)$$

[29] $\psi' = \psi$ if ψ is homogeneous.

is identically null. Consequently, according to Lemma 3, $\psi' \circ \varepsilon \circ \nu$ can be factored by $X_1 + \ldots + X_\kappa$. Thus, according to Lemma 5, $\psi' \circ \varepsilon$ can be factored by

$$\varphi(Z) = \sum_{\ell=1,\ldots,\kappa} V_{2\ell-1} \prod_{\ell' \neq \ell} V_{2\ell'}$$

By noticing that

$$\varphi \circ \delta = \phi_0$$

$\psi' \circ \varepsilon \circ \delta$ and thus $\Delta^t \cdot \psi$ can be factored by ϕ_0. As $\gcd(\Delta, \phi_0) = 1$, ψ is a multiple of ϕ_0.

□

H.3 Proof of Proposition 8

Without loss of generality, we prove here that there does not exist any polynomial-size $\{+, -, \times\}$-circuit representing a polynomial ϕ satisfying $\phi \circ \alpha = \phi_0$. The extension to multiples of ϕ_0 is not difficult (but not straightforward).

According to notation of Definition 7, we consider the tuples $V = (V_1, \ldots, V_{2\kappa})$ and $Y = ((X_{i\ell}, R_{i\ell})_{i=1,\ldots,t}, (S_{2\ell-1,i}, S_{2\ell,i})_{i=1,\ldots,2\kappa})_{\ell=1,\ldots,\kappa}$. We enhance the power of the attacker by letting it choose the Δ_κ-symmetric polynomials $\alpha_1, \ldots, \alpha_{\gamma'}$.

As $\phi \circ \alpha(Y) = \phi_0(Y)$, the equality also holds by setting $X_{i\ell} = R_{i\ell} = 1$ for any $i = 1, \ldots, t$ and $S_{2\ell-1,i} = S_{2\ell,i}$ for any i, ℓ. We then consider the polynomials $\nu_1, \ldots, \nu_\gamma$ and ψ defined over $V, S = ((S_{\ell,i})_{i=1,\ldots,2\kappa})_{\ell=1,\ldots,\kappa}$ by

$$\begin{aligned}
\nu_i(S) &= \alpha_i \left((1, \ldots, 1, (S_{2\ell,i}, S_{2\ell,i})_{i=1,\ldots,2\kappa})_{\ell=1,\ldots,\kappa} \right) \\
\psi(S, V) &= \frac{1}{\kappa} \phi_0 \left((1, \ldots, 1, (S_{2\ell,i}, S_{2\ell,i})_{i=1,\ldots,2\kappa})_{\ell=1,\ldots,\kappa}, V \right) \\
&= \prod_{\ell=1,\ldots,\kappa} \left(\sum_{i=1}^{2\kappa} S_{2\ell,i} V_i \right)
\end{aligned}$$

Similarly to the definition of α, we consider the function

$$\nu(S, V) = (\nu_1(S), \ldots, \nu_{\gamma'}(S), V)$$

To establish our result, it suffices to show that there does not exist any (polynomial-size) polynomial ϕ such that $\phi \circ \nu = \psi$. To achieve it, we first notice that that the polynomials ν_i remain Δ_κ-symmetric. Without loss of generality, we will assume that the polynomials ν_i are homogeneous (otherwise we split them into homogeneous polynomials). Moreover, as $\deg \psi = \kappa$, one can assume that $\deg \nu_i \leq \kappa$. Consider the two sets I_1, I_2 defined by

- $I_1 = \{i \in \{1, \ldots, \gamma'\} | \deg \nu_i < \kappa\}$,
- $I_2 = \{i \in \{1, \ldots, \gamma'\} | \deg \nu_i = \kappa\}$

Let

$$V_\kappa \stackrel{\mathsf{def}}{=} \{\boldsymbol{v} \in \{0,1\}^{2\kappa} | v_1 + \cdots + v_{2\kappa} = \kappa\}$$

For a given $\boldsymbol{v} \in \mathbb{Z}_n^{2\kappa}$, the polynomial ψ_v is defined by,

$$\psi_{\boldsymbol{v}=(v_1,\ldots,v_{2\kappa})}(S) = \psi(S,\boldsymbol{v}) = \cdot \prod_{\ell=1,\ldots,\kappa} \left(\sum_{i=1}^{2\kappa} v_i S_{2\ell,i} \right)$$

Lemma 8. *Let $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_r \in V_\kappa$ and $a_1,\ldots,a_r \in \mathbb{Z}_n \setminus \{0\}$. The polynomial $a_1\psi_{\boldsymbol{v}_1} + \ldots + a_r\psi_{\boldsymbol{v}_r}$ cannot be written as a polynomial $p((\nu_i)_{i\in I_1})$.*

Proof. By Lemma 4, one can straightforwardly show that $\psi_{(1,0,\ldots,0)}$ ($\psi_{(1,0,\ldots,0)}(S) = s_{2,1}s_{4,1}\cdots s_{2\kappa,1}$) cannot be written as a polynomial $p((\nu_i)_{i\in I_1})$. Given $\tau \in \mathbb{Z}_n^{2\kappa}$, we denote by $\nu_1^\tau,\ldots,\nu_{\gamma'}^\tau$ the polynomials $\nu_1,\ldots,\nu_{\gamma'}$ where the variables $s_{2\ell,i}$ are substituted by $\tau_i s_{2\ell,1}$ for any $1 \le i \le 2\kappa$ and φ_i denotes the polynomial $\psi_{\boldsymbol{v}_i}$ by doing the same substitution. It is important to notice that $\nu_1^\tau,\ldots,\nu_{\gamma'}^\tau$ are symmetric polynomials defined over $s_{2,1}, s_{4,1}, \cdots, s_{2\kappa,1}$.

Moreover, $\sum_{i=1}^r a_i\varphi_i(s_{2,1}, s_{4,1}, \cdots, s_{2\kappa,1}) = q(\tau)s_{2,1}s_{4,1}\cdots s_{2\kappa,1}$ where q is a degree-κ polynomial. Clearly, q is not null because. Indeed, by definition of V_κ, each φ_i contains at least one monomial which does not belong to the other polynomials $\varphi_{j\neq i}$.

Thus, according to the famous lemma of Schwartz and Lippel [Sch80], $q(\tau) = 0$ with negligible probability over the choice of τ. Let τ^* such that $q(\tau^*) \neq 0$. The equality $p((\nu_i)_{i\in I_1}) = a_1\psi_{\boldsymbol{v}_1} + \ldots + a_r\psi_{\boldsymbol{v}_r}$ implies that $p((\nu_i^{\tau^*})_{i\in I_1}) = C \cdot s_{2,1}\cdots s_{2\kappa,1}$ with $C \neq 0$ contradicting Lemma 4.

□

The result is a direct consequence of this lemma. Given a polynomial ϕ, we consider the polynomial ϕ_v defined by $\phi_{\boldsymbol{v}}(\nu_1,\ldots,\nu_{\gamma'}) = \phi \circ \nu$. Let us assume that $\phi \circ \nu = \psi$ implying that for each $\boldsymbol{v} \in V_\kappa$, $\psi_v = \phi_{\boldsymbol{v}}$.

Because $\deg\psi = \kappa$, we can write $\phi_{\boldsymbol{v}}(\nu_1,\ldots,\nu_t) = \phi'_{\boldsymbol{v}}(\nu_{i\in I_1}) + \phi''_{\boldsymbol{v}}(\nu_{i\in I_2})$ with $\deg\phi''_{\boldsymbol{v}} = 1$. As $|I_2| \le \gamma'$ is polynomial but not $\#V_\kappa$, there exist $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_r \in V_\kappa$ s.t. the linear functions $\phi''_{\boldsymbol{v}_1},\ldots,\phi''_{\boldsymbol{v}_r}$ are linearly dependant. It follows that there exist $a_1,\ldots,a_r \in \mathbb{Z}_n \setminus \{0\}$ such that

$$a_1\phi''_{\boldsymbol{v}_1}(\nu_{i\in I_2}) + \ldots + a_r\phi''_{\boldsymbol{v}_r}(\nu_{i\in I_2}) = 0$$

It implies that $a_1\phi'_{\boldsymbol{v}_1}(\nu_{i\in I_1}) + \cdots + a_r\phi'_{\boldsymbol{v}_r}(\nu_{i\in I_1}) = a_1\psi_{\boldsymbol{v}_1} + \ldots + a_r\psi_{\boldsymbol{v}_r}$ contradicting Lemma 8.

□