IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

The Role of Non-Positional Arithmetic on Efficient Emerging Cryptographic Algorithms

PAULO MARTINS[1], (Member, IEEE), LEONEL SOUSA[1], (Senior Member, IEEE)
[1]INESC-ID, Instituto Superior Técnico, Universidade de Lisboa

Corresponding author: Paulo Martins (e-mail: paulo.sergio@netcabo.pt).

This work was supported by national funds through FCT, Fundação para a Ciência e a Tecnologia, under project UIDB/50021/2020 and by EU's Horizon 2020 research and innovation programme under grant agreement No. 779391 (FutureTPM).

ABSTRACT Number representation systems establish ways in which numbers are mapped to computer architectures, and how operations over the numbers are translated into computer instructions. The efficiency of public-key cryptography is strongly affected by the used number representations, as these systems are constructed from mathematically inspired problems to ensure security, and thus rely on operations over large integers. In this paper, unconventional representations systems, including the Residue Number System (RNS) and stochastic number representations, are systematically reviewed. Homomorphic representations, which allow for parties to operate on data without having access to their plaintext representation, are also considered. The main goal of this survey is to introduce the reader to key aspects of non-traditional number representations that may be exploited for public-key cryptography, without delving too much into the details. Examples of the methods and algorithms herein surveyed include subquadratic modular multiplication for isogeny-based cryptography, the acceleration of Goldreich-Goldwasser-Halevi (GGH) decryption by an order of magnitude, the improvement of the Direct Anonymous Attestation (DAA) protocol both in terms of storage requirements and the time taken to execute it, and efficient algorithm-hiding Fully Homomorphic Encryption (FHE). The implementation of this type of systems in both sequential and parallel platforms is analysed, and their performance is compared with traditional approaches. We hope this work sows the seed of further research on the application of non-positional number arithmetic to other cryptographic use-cases.

INDEX TERMS Homomorphic Encryption, Lattice-based Cryptography, Modular Arithmetic, Residue Number System (RNS), Stochastic Computing

I. INTRODUCTION

PUBLIC-KEY cryptography has become the root of security on devices ranging from high-performance platforms to Internet of Things (IoT) systems with restricted capabilities [1]. In public-key cryptographic systems, each user creates a pair of keys. The public-key is widely distributed, while the other, that is private, should be protected from data disclosure. Depending on the system, the pair of keys may have different uses. A straightforward use corresponds to the ciphering of messages with the public-key, and the deciphering of the corresponding cryptogram with the private-key [2], [3]. Another use is related to the generation and verification of digital signatures: one message might be exclusively associated with a user if they produce a signature of that message under their private-key [2], [3]. Any other user might confirm this association through the verification of the signature with the public-key. A more sophisticated use allows for the processing of encrypted data. A user might encrypt data, and send the resulting cryptograms to a server. The server may process the cryptograms with the help of the public-key and return the encrypted results. After having deciphered the message, the user will have access to the results of the computation as if he or she had processed the data himself or herself [4], [5], [6].

Hard mathematical problems are used to ensure the security of public-key cryptographic systems. These systems are designed so that the derivation of the private-key from the public-key is as hard as computing the solution to problems like the discrete logarithm [3], [7], [8], the factorisation of large integers [2], [9], or the determination of the shortest

vector of a lattice [10]. The number of steps required to solve these problems is evaluated as a function of the length of the input. Then, the dimension of the keys is enlarged in a way that makes the system impossible to break while it is under use. Since the keys often need to remain valid for a long time, in some cases several years, their dimension is made very large, leading to the need of processing sizeable mathematical structures in cryptographic operations, like encryption or signing.

Innovative arithmetic for the implementation of cryptographic operations has been fundamental to achieve systems capable of both taking advantage of the architectural developments of high-performance processors, while also executing on platforms with limited memory and processing power. While in the past these goals were achieved with the application of the RNS to number-theoretic cryptography [11], [12], [13], if a large quantum computer is ever developed this type of systems will no longer be secure [14]. As a result of the National Institute of Standards and Technology (NIST) efforts to standardise post-quantum cryptography [15], a large number of cryptosystems has been proposed. Security is built upon problems related to lattices and isogenies, among others, whose arithmetic structures have yet to be deeply investigated [16], [17]. Recent developments of cryptography have also enabled the processing of encrypted data, but they offer a set of instructions that is radically different from traditional programming [4], [5], [6]. Therefore, there is an urgent need to develop novel arithmetic methods and algorithms supporting emerging cryptography, and achieve more efficient, secure and usable systems.

This paper adopts a tutorial style to introduce unconventional number representations, including homomorphisms, and how they can be leveraged to achieve more efficient, usable and robust cryptosystems. Table 1 includes the number representations described in this paper, and associates them with cryptosystems that benefit from their usage. Moreover, it includes references to the Sections in which the representation systems are described, and where they are applied to public-key cryptographic schemes.

The remainder of this paper is organised as follows. Section II reviews non-conventional arithmetic systems, describing how their properties differ from traditional binary representation systems. Lattices are a recurrent topic throughout the article, being used not only to support the security of cryptosystems, but also to aid the design of novel representation systems. Hence, they are introduced in Section III. In Section IV, state-of-the-art RNS techniques are presented for improving the decryption of the lattice-based GGH cryptosystem. Recently proposed methods for modular multiplication are reviewed in Section V which reduce the complexity of standardised Elliptic Curve Cryptography (ECC) and isogeny-based cryptography. Furthermore, methods are described in Section VI that methodically design homomorphic circuits based on stochastic number representations, while efficiently achieving secrecy of the processing algorithm. Rather than trying to provide algorithm-secrecy in general, this paper focuses on a subset of applications that can be efficiently evaluated homomorphically. Moreover, Section VII considers the representation of values through homomorphisms. This allows for one party to provide their data to another party, while preserving its confidentiality. This property is exploited to reduce the computational complexity of a lattice-based DAA scheme [18]. Techniques for the efficient implementation of the aforementioned techniques, alongside an experimental evaluation of their effectiveness, are presented in Section VIII. Finally, conclusions are drawn in Section IX and directions for future research are discussed.

II. UNCONVENTIONAL NUMBER REPRESENTATIONS

In this section, unconventional number representations are reviewed in a systematic way so as to identify their key properties to be used on cryptography. We start by giving intuitive definitions of what number representation systems and their underlying computing models are.

Definition 1 (Number representation system)**.** Number representation systems are associated with real-life applications. Each application defines a set of elements and describes operations to process them. A number representation system is a morphism mapping elements from the set that arises naturally when describing the application onto another set that may be interpreted by an underlying computing model. Note that this mapping need not be only a function of the original set; in certain cases, for instance, randomness may be added as a second variable, thus associating each element of the original set onto multiple of the codomain. Moreover, it maps the operations necessary for the processing of the application over the original set onto operations, or combinations thereof, compatible with the underlying computing model.

Definition 2 (Computing model)**.** A computing model defines admissible ways to represent numbers, and the operations that are available to process them. It is often used to abstract physical implementations of a processor.

An example of a traditional number representation system, depicted in Figure 1, is the binary representation system that maps elements from the finite field $\mathbb{F}_p$ with a prime order $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$ onto arrays of 8 32-bit integers, which might be processed by a 32-bit processor. Addition over $\mathbb{F}_p$ may be mapped onto an algorithm over $(\mathbb{Z}_{2^{32}})^8$ that iteratively computes $2^{32}c_i + z_i = x_i + y_i + c_{i-1}$, where x_i, y_i are the words of the input, z_i the words of the output, and c_i correspond to the carries that must be propagated from the least significant digit to the most significant one. Moreover, one would need to subtract p from the result, if the output were greater than or equal to p. Multiplications could be mapped to an application of the Montgomery algorithm [19], which we will have a chance to review in the context of other number representations.

While Definition 2 has been traditionally applied to abstract physical implementations of processors, in this article it will be used more generically, and may:

Number representation	Residue Number System		Stochastic Computing	Homomorphisms
introduced in Section	II-A		II-B	II-C
applied to	GGH	Isogeny	FHE	DAA
in Section	IV	V	VI	VII

TABLE 1: Surveyed numeral representation systems, along with the schemes for which methods/algorithms are developed exploiting them

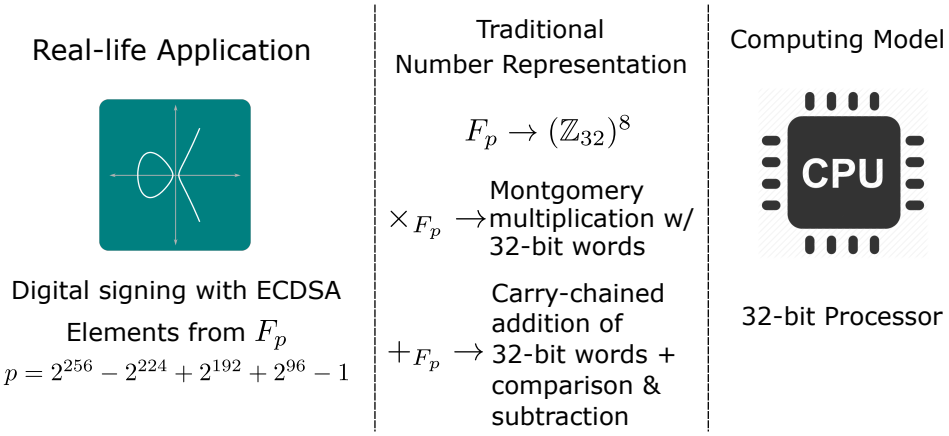


FIGURE 1: Example of traditional number representation applied to digital signatures

- represent a third party, such that admissibility of representations excludes any number representation system which would allow the third party to deduce the element of the original domain to which that representation corresponds;
- represent an homomorphic encryption system, such that the ways to represent numbers and process them are the ones that arise from the mathematical structure of that encryption system.

A. RESIDUE NUMBER SYSTEM

The Chinese Remainder Theorem (CRT) states that for co-prime integers $b_{1,0}, \ldots, b_{1,h_1-1}$ and for $B_1 = b_{1,0} \times \ldots \times b_{1,h_1-1}$, the ring $\mathbb{Z}_{B_1}$ is isomorphic to $\mathbb{Z}_{b_{1,0}} \times \ldots \times \mathbb{Z}_{b_{1,h_1-1}}$ with the following map:

$$\begin{aligned} g &: \mathbb{Z}_{B_1} \rightarrow \mathbb{Z}_{b_{1,0}} \times \ldots \times \mathbb{Z}_{b_{1,h_1-1}} \\ g(a) &= (a_{1,0}, \ldots, a_{1,h_1-1}) \\ &= (a \bmod b_{1,0}, \ldots, a \bmod b_{1,h_1-1}) \end{aligned}$$

and inverse

$$a = \left[\sum_{i=0}^{h_1-1} \xi_{1,i} \frac{B_1}{b_{1,i}} \right]_{B_1} = \sum_{i=0}^{h_1-1} \xi_{1,i} \frac{B_1}{b_{1,i}} - \alpha B_1 \quad (1)$$

where $\xi_{1,i} = \left[a_{1,i} \frac{b_{1,i}}{B_1} \right]_{b_{1,i}}$.

The RNS exploits the CRT to replace additions, subtractions and multiplications over large integers in $\mathbb{Z}_{B_1}$ by the coefficient-wise additions, subtractions and multiplications over the smaller channels $\mathbb{Z}_{b_{1,0}}, \ldots, \mathbb{Z}_{b_{1,h_1-1}}$, which may be efficiently processed by modern processors. While these operations are made faster with the RNS, operations such as divisions and modular reductions are harder to implement. One often has to use basis extensions to deal with these operations. This procedure exploits (1) to extend the representation of a number in basis $\mathcal{B}_1 = \{b_{1,0}, \ldots, b_{1,h_1-1}\}$ to another basis $\mathcal{B}_2 = \{b_{2,0}, \ldots, b_{2,h_2-1}\}$. In cases where an error of αB_1 can be tolerated, extensions may be performed with FastBConv to approximate (1) in an efficient way [20]:

$$a_{2,i} = \text{FastBConv}(a, \mathcal{B}_1) = \sum_{i=0}^{h_1-1} \xi_{1,i} \frac{B_1}{b_{1,i}} \bmod b_{2,i} \quad (2)$$

When such an error cannot be tolerated, an extra residue $a_{\mathsf{sk}} = a \bmod b_{\mathsf{sk}}$ may be computed. This enables the computation of α as

$$\alpha = \left[(\text{FastBConv}(a, \mathcal{B}_1) - a_{\mathsf{sk}}) B_1^{-1} \right]_{b_{\mathsf{sk}}}$$

for $|a| < \lambda B_2$, an integer λ and $b_{\mathsf{sk}} \geq 2(h_2 + \lambda)$ [21, Lemma 6]. In this case, the basis extension may be terminated with FastBConvSK:

$$a_{2,i} = \text{FastBConvSK}(a, \mathcal{B}_1, \alpha) = \sum_{i=0}^{h_1-1} \xi_{1,i} \frac{B_1}{b_{1,i}} - \alpha B_1 \bmod b_{2,i} \quad (3)$$

B. STOCHASTIC COMPUTING

A stochastic representation of a number $x \in [0,1]$ is defined to be a sequence of n bits, $x_1, \ldots, x_n$, drawn from a Bernoulli distribution, such that the probability $P(x_i = 1) = x, \forall_{1 \leq i \leq n}$ [22]. AND and XOR operations can be combined to perform the following two operations on three independent stochastic representations $x, y, s \in [0,1]$ (note that when two representations are not statistically independent, they can be rotated to become so):

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

$$z_i = x_i \wedge y_i \Rightarrow z = xy \tag{4}$$

$$z_i = ((1 \oplus s_i) \wedge x_i) \oplus (s_i \wedge y_i) \Rightarrow z = (1-s)x + sy \tag{5}$$

where $\wedge$ and $\oplus$ stand for the AND and XOR operations, respectively.

These two operations serve as the basis for the polynomial evaluation procedure described in Algorithm 1.

Algorithm 1 De Casteljau's algorithm for the evaluation of a polynomial in Bernstein form [23]

Require: $B(X) = \sum_{i=0}^{d} \binom{d}{i} b_i X^i (1-X)^{d-i}$
Require: x_0
1: **for** $i \in \{0, \ldots, d\}$ **do**
2: $\quad b_i^{(0)} = b_i$
3: **end for**
4: **for** $j \in \{1, \ldots, d\}$ **do**
5: $\quad$ **for** $i \in \{0, \ldots, d-j\}$ **do**
6: $\quad\quad b_i^{(j)} = b_i^{(j-1)}(1-x_0) + b_{i+1}^{(j-1)} x_0$
7: $\quad$ **end for**
8: **end for**
9: **return** $B(x_0) = b_0^{(d)}$

C. HOMOMORPHISMS

A traditional computing model considers programs that operate, instruction by instruction, over imputed data. Alternatively, when distributed computing is considered, some of these inputs might belong to another party. In the former computing model, isomorphisms, like the RNS, provide for ways to manipulate data in more computationally efficient domains (e.g. RNS channels), with operations having counterparts in the original domain (e.g. large integer arithmetic). In the latter computing model, values should be represented in a way that prevents the second party from deriving information about the input. In this case, the parties might exploit homomorphisms to protect the confidentiality of their data.

Definition 3 (Homomorphism)**.** A map $f : A \to B$ is said to be homomorphic with respect to the operator μ of arity k, defined on both A and B, if $f(\mu_A(a_1, \ldots, a_k)) = \mu_B(f(a_1), \ldots, f(a_k))$ for all elements $a_1, \ldots, a_k \in A$.

A basic example of homomorphism corresponds to Baum et al's commitment scheme [24]. Let $\mathcal{R} = \mathbb{Z}[X]/\langle \Phi_m(X) \rangle$ be the set of polynomials equipped with addition and multiplication modulo the m$^{\text{th}}$ cyclotomic polynomial, $\Phi_m(X) = \prod_{\substack{1 \le k \le m \\ \gcd(k,m)=1}} (X - e^{2i\pi\frac{k}{m}})$ – note that even though the formula for $\Phi_m(X)$ involves complex numbers, it is a polynomial with integer coefficients. Furthermore, let $\mathcal{R}_q = \mathcal{R}/(q\mathcal{R})$ be $\mathcal{R}$ with coefficients reduced modulo an integer q. With this scheme, the addition of commitments to messages $\vec{s}_1, \vec{s}_2 \in \mathcal{R}_q^d$, produces a commitment to $\vec{s}_1 + \vec{s}_2$. More complex examples of homomorphism include the FHE scheme described in Section VI-A.

Definition 4 (Baum et al's Commitment Scheme)**.** Baum et al's commitment scheme is a triplet of functions (KeyGen, Commit, Open) defined as follows.

- C.KeyGen(k): Given a security parameter k, generates the system parameters $(q,\ \mathcal{R}_q,\ \alpha,\ \gamma,\ \vec{b})$, where q is a prime modulus defining $\mathcal{R}_q$, α and γ are positive numbers, and B is a uniformly random matrix of polynomials in $\mathcal{R}_q^{k \times (d+1)}$, for some positive integer d.
- Commit($\vec{s}$): To commit to a message $\vec{s} \in \mathcal{R}_q^d$, choose a uniformly random vector of invertible polynomials $\vec{r} \in \mathcal{D} \subseteq \mathcal{R}^k$ such that $\|\vec{r}\|_\infty \le \alpha$. Compute $\vec{c} = \mathsf{COM}(\vec{s}, \vec{r}) = \vec{r}B + (0, \vec{s})$, and output $\vec{c}$.
- Open($\vec{c}, \vec{s}, \vec{r}, \boldsymbol{p}$): A valid opening of a commitment $\vec{c}$ is a 3-tuple: $\vec{s} \in \mathcal{R}_q^d$, $\vec{r} \in \mathcal{R}^k$ and an invertible polynomial $\boldsymbol{p} \in \mathcal{R}$ such that $\|\boldsymbol{p}\|_\infty \le \gamma$. The verifier checks that

$$\vec{r}B + (\mathbf{0}, \boldsymbol{p}\vec{s}) = \boldsymbol{p}\vec{c} \text{ with } \|\vec{r}\|_\infty \le \alpha.$$

The security of the scheme in Definition 4 is based on the difficultness of solving the Ring Learning with Errors (RLWE) problem, which will be described in Section III.

III. LATTICES

Lattices have significant applications in the field of cryptography [25], which include ensuring the security of cryptographic schemes like the one in Definition 4, and supporting the design of number representations, such as the one in Section V. A lattice can be seen both as an additive subgroup of $\mathbb{R}^m$, or as the vector space generated by all linear combinations with integer coefficients of a set $\mathcal{R} = \{\vec{r}_0, \ldots, \vec{r}_{n-1}\}$, with $\vec{r}_i \in \mathbb{R}^m$, of linearly independent vectors:

$$\mathcal{L}(\mathcal{R}) = \left\{ \sum_{i=0}^{n-1} z_i \vec{r}_i : z_i \in \mathbb{Z} \right\} \tag{6}$$

The set $\mathcal{R}$ can also be associated with a matrix R having the vectors $\vec{r}_i$ as rows. The previous expression can thus be written as $\mathcal{L}(R) = \{\vec{z} \times R : \vec{z} \in \mathbb{Z}^n\}$. It is said that the rank of the lattice is n and its dimension is m. Each lattice induces an equivalence relation over span(R) (i.e. the linear space generated by the rows of R). Two vectors in span(R) are congruent if their difference is in $\mathcal{L}(R)$:

$$a = b (\mathrm{mod}\ \mathcal{L}(R)) \Leftrightarrow a - b \in \mathcal{L}(R) \tag{7}$$

We associate with each basis the parallelepiped:

$$\mathcal{P}(R) = \left\{ \sum_{i=0}^{n-1} w_i \vec{r}_i : w_i \in \left(-\frac{1}{2}, \frac{1}{2} \right] \right\} \tag{8}$$

For any point $\vec{y} = \vec{w} + \vec{x} \in$ span(R), where $\vec{w} \in \mathcal{L}(R)$ and $\vec{x} \in \mathcal{P}(R)$, the reduction of $\vec{y}$ modulo $\mathcal{P}(R)$ is defined as $\vec{x} = \vec{y} \bmod \mathcal{P}(R)$. It should be noted that lattices have an infinite number of bases for $n \ge 2$. The modular reduction has a different meaning for each basis, since they are associated with different parallelepipeds. An example of this is featured in Figure 2, where the point $\vec{c}$ is reduced both

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

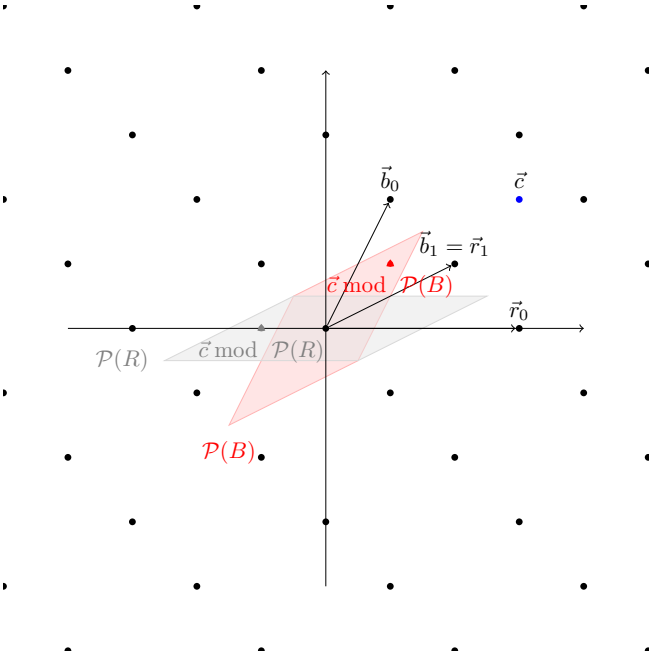


FIGURE 2: Two basis $\vec{r}_0, \vec{r}_1$ and $\vec{b}_0, \vec{b}_1$ of the same lattice, along with the corresponding parallelepipeds in red and grey, are represented. The point $\vec{c}$ is reduced modulo the two parallelepipeds

modulo $\mathcal{P}(R)$ and $\mathcal{P}(B)$, producing two different points, which are represented as triangles, whilst $\mathcal{L}(R) = \mathcal{L}(B)$.

Any two basis of the same lattice, B and R, are related by a unimodular matrix (a square integer matrix having determinant $+1$ or -1):

$$B = UR. \tag{9}$$

The volume of a lattice $\mathcal{L}(R)$ is given by:

$$\text{vol}(\mathcal{L}(R)) = \sqrt{\det(RR^T)}. \tag{10}$$

From (9) and (10), one concludes that the volume of a lattice is independent of the basis used to describe it.

Theorem 1 (Minkowski's First Theorem [26])**.** *For any full-rank lattice $\mathcal{L}$ of rank n, the norm of its shortest vector $\lambda_1(\mathcal{L})$, satisfies*

$$\lambda_1(\mathcal{L}) \leq \sqrt{n}\, \sqrt[n]{\det(\mathcal{L})}$$

Lattice-based Cryptography (LBC) is usually supported on either the Closest Vector Problem (CVP), the Shortest Vector Problem (SVP) or the General Learning with Errors (GLWE), the definitions of which follow [10], [27], and can be instantiated for any norm. The Learning with Errors (LWE) and RLWE problems are captured in Definition 7, and correspond to GLWE when $m = 1$ and $n = 1$, respectively.

Definition 5 (CVP)**.** Given a base $R \in \mathbb{R}^{n \times m}$, and $\vec{y} \in \mathbb{R}^m$, find $\vec{x} \in \mathcal{L}(R)$, such that $||\vec{y} - \vec{x}|| = \min_{\vec{z} \in \mathcal{L}(B)} ||\vec{y} - \vec{z}||$.

Definition 6 (SVP)**.** Given a base $R \in \mathbb{R}^{n \times m}$, find $\vec{x} \in \mathcal{L}(R)$, such that $||\vec{x}|| = \min_{\vec{z} \in \mathcal{L}(B) \setminus \vec{0}} ||\vec{z}||$.

Definition 7 (GLWE)**.** Let $n, m, q \in \mathbb{Z}$; let $\mathcal{R} = \mathbb{Z}[X]/\langle \Phi_m(X) \rangle$, $\mathcal{R}_q = \mathcal{R}/(q\mathcal{R})$; and let χ be a distribution over $\mathcal{R}$ (typically a Gaussian distribution). Given arbitrarily many samples $(\vec{x}_i, y_i) \in \mathcal{R}_q^{n+1}$, where $y_i = \langle \vec{x}_i, \vec{s} \rangle + e_i$, with $\vec{x}_i, \vec{s} \leftarrow \mathcal{R}_q^n$ sampled uniformly and $e_i \leftarrow \chi$, find $\vec{s}$.

In Definition 7, $\Phi_m(X)$ refers to the m$^{\text{th}}$ cyclotomic polynomial: $\Phi_m(X) = \prod_{\substack{1 \leq k \leq m \\ \gcd(k,m)=1}} (X - e^{2i\pi \frac{k}{m}})$. We may intuitively establish a connection between the LWE problem and lattices (a similar rationale can be conducted for the RLWE). First the lattice $\mathcal{L}(R)$ is defined, where the matrix $R \in \mathbb{Z}^{n \times t}$ has t $\vec{x}_i$ samples as columns. If one was able to compute the closest vector, $\vec{y'}$ to $\vec{y} = (y_0, \ldots, y_{t-1})$ in the lattice, then solving the system $\vec{s} \times R = \vec{y'}$ would provide a solution to the LWE problem.

IV. A RNS BASED GGH CRYPTOSYSTEM

A main operation limiting the performance of post-quantum cryptography, namely GGH, corresponds to the division and rounding used to approximate the closest lattice point to a vector. Division and rounding has no parallel in traditional cryptography. This issue has been addressed in [28], [29], [30]. Herein, the techniques proposed therein are reviewed, which allow for the exploitation of unconventional number representations to adapt GGH to modern computing platforms featuring data and multicore parallelism.

A. GGH CRYPTOSYSTEMS

For the remaining of this paper, $[\cdot]_q$ denotes the centred residue modulo q in $[-q/2, q/2)$, $|\cdot|_q$ the traditional residue modulo q in $[0, q)$, and $\lfloor \cdot \rceil$ rounding to the nearest integer.

Encryption, with [31], consists of adding a small "error", which represents the message, to a vector of the lattice. The public key corresponds to a "bad basis" of the lattice, with which solving the CVP is hard. Contrastingly, the private key is a "good basis", enabling the computation of the closest lattice vector for small errors. To obtain the original message, the receiver computes the closest lattice point to the cryptogram, and produces the difference between the two.

Babai's Round-Off [32] provides approximate solutions to the CVP. It is supported on the observation that $\text{span}(R)$ is equal to the disjoint union of the parallelepiped $\mathcal{P}(R)$ shifted by lattice points $\vec{u} \in \mathcal{L}(R)$:

$$\text{span}(R) = \bigcup_{\vec{u} \in \mathcal{L}(R)} \vec{u} + \mathcal{P}(R) \tag{11}$$

Hence, an approximation of the closest lattice point to $\vec{v} \in \text{span}(R)$ corresponds to determining to which tile $\vec{u} + \mathcal{P}(R)$ $\vec{v}$ belongs to and returning $\vec{u} \in \mathcal{L}(R)$. The more orthogonal the basis R is, the better the results will be.

[31] is supported on integer full-rank lattices (i.e. $\vec{r_i} \in \mathbb{Z}^m$ and $m = n$ in (6)). The private basis is produced as a rotated nearly-orthogonal basis, such that Babai's Round-Off procedure may be used to compute the closest vector [32]. Moreover, the public basis is of a Hermite Normal Form

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

(HNF). The HNF is a lower triangular basis of $\mathcal{L}(R)$, $H \in \mathbb{Z}^{n\times n}$, such that:

$$\forall\, 0 \leq i,j < n, \quad 0 \leq H_{i,j} \begin{cases} = 0 & \text{if } i < j \\ \geq 1 & \text{if } i = j \\ < H_{j,j} & \text{if } i > j \end{cases} \tag{12}$$

This normal form is unique, and can be efficiently computed from any basis of $\mathcal{L}$. If an efficient attack based on the HNF of a lattice was produced, any public-key could be broken, regardless of its form, because it would be possible to efficiently compute the HNF from it. In contrast, if an attack was developed based on a different basis of the lattice, there would be no guarantee that that basis could be efficiently computed from the HNF, making the HNF a good choice for the public-key. In particular, Rose's cryptosystem uses bases of an Optimal Hermite Normal Form (OHNF) as the public-key. OHNFs form a subclass of HNFs, where all but the first column are trivial. More formally, H is an OHNF basis of $\mathcal{L}$ if and only if H is an HNF basis and $\forall_{0<i<n}, H_{i,i} = 1$. As an example, most lattices with a prime number as the determinant have an OHNF.

The key generation step consists of finding a rotated nearly-orthogonal private basis, such that its HNF H is optimal. Since H is of an OHNF, the public-key reduces to a column, denoted as $H0$. A plaintext is then represented as a vector $\vec{p} \in \mathbb{Z}^n$. To encrypt it, $\vec{p}$ is reduced modulo the public basis, by applying Algorithm 2. Due to the public basis structure, the cryptogram corresponds to a vector, where all the entries but the first are zero. Therefore, it suffices a scalar c to store its value.

Algorithm 2 Rose's Encryption Algorithm

Require: $p \in \mathbb{Z}^n$, $H0 \in \mathbb{Z}^n$
Ensure: $c \in \mathbb{Z}$
1: $c \leftarrow p[0]$
2: **for** $i \leftarrow n-1$ **to** 1 **do**
3: $\quad c \leftarrow c - p[i] \times H0[i]$
4: **end for**
5: $c \leftarrow c \bmod H0[0]$
6: **return** c

In order to decipher $\vec{c}$, Babai's Round-Off algorithm can be applied. This procedure gives an approximation to the CVP, as represented in Algorithm 3.

Algorithm 3 Rose's Decryption Algorithm

Require: $\vec{c} \in \mathbb{Z}$, $R_0^{-1} \in \mathbb{Q}^n$, the first row of R^{-1}
Ensure: $\vec{p} \in \mathbb{Z}^n$
1: $\vec{p} \leftarrow (c, 0, \ldots, 0) - \left\lfloor cR_0^{-1} \right\rceil R$
2: **return** $\vec{p}$

B. DIVISION AND ROUNDING

Algorithm 3 can be modified to make it more suitable for implementation with the RNS. The operation $\left\lfloor \vec{c}R^{-1} \right\rceil$ in Algorithm 3 is replaced by the approximation $\vec{\mu}$ of $\lfloor \gamma\vec{c}R^{-1} \rfloor$ through an RNS Montgomery reduction [19], where $\vec{c} = (c, 0, \ldots, 0)$. The scaling by γ enables the detection and correction of the errors resulting from the approximate RNS Montgomery reduction: for an encryption $\vec{c} = \vec{k}R + \vec{p}$ of $\vec{p}$, the computation of $\vec{\mu} \approx \lfloor \gamma\vec{c}R^{-1} \rfloor$ produces the vector $\gamma\vec{k}$ plus a small error term that is nonzero modulo γ. We retrieve the error term from the centred residue of $\vec{\mu}$ modulo γ, and then decrypt the message with $\vec{p} = \vec{c} - \frac{\vec{\mu} - [\vec{\mu}]_\gamma}{\gamma} R = \vec{c} - \vec{k}R \bmod m_\sigma$.

The resulting procedure, depicted in Algorithm 4, is based on the rewriting of $\lfloor \gamma\vec{c}R^{-1} \rfloor$ using integer arithmetic as

$$\lfloor \gamma\vec{c}R^{-1} \rfloor = \frac{\gamma\vec{c}\hat{R} - |\gamma\vec{c}\hat{R}|_d}{d} \tag{13}$$

where $\hat{R} = R^{-1}d$ and $d = \det(R)$. While R^{-1} is not typically an integer matrix, $\hat{R}$ is. Initially, $\vec{q}_1 = -cR'_0d^{-1} \bmod B_1$ is computed in an RNS basis $\mathcal{B}_1$, and approximately extended to a second basis $\{m_\sigma, \gamma\}$. Notice that in Algorithm 4 $\vec{w}$ satisfies

$$\vec{w} = \frac{\vec{c}R' + \vec{q}_2 d}{B_1} = |\gamma\vec{c}\hat{R}|_d + \vec{e}d \bmod \{m_\sigma, \gamma\} \tag{14}$$

since $\vec{c}R' + \vec{q}_2 d = \vec{c}R' - \vec{c}R'd^{-1}d + \vec{\alpha}B_1 d = 0 \pmod{B_1}$, where $\vec{\alpha}$ is an error term associated with an approximate basis extension. Hence, B_1 divides $\vec{c}R' + \vec{q}_2 d$ over the integers. Furthermore, $\frac{\vec{c}R' + \vec{q}_2 d}{B_1} = \vec{c}\gamma\hat{R}B_1B_1^{-1} = \gamma\vec{c}\hat{R} \pmod d$. The value of B_1 is chosen to ensure that $\vec{e}$ is small. More concretely, for a $d < \epsilon B_1$, we have that $\|\vec{e}\| < \epsilon + h_1$. This enables the production of $\vec{\mu} = \lfloor \gamma\vec{c}R^{-1} \rfloor - \vec{e} = \left(\gamma\vec{c}\hat{R} - |\gamma\vec{c}\hat{R}|_q - \vec{e}d\right)/d \bmod \{m_\sigma, \gamma\}$ that approximates (13). By noticing that

$$\vec{\mu} = \gamma\vec{k} + \lfloor \gamma\vec{p}R^{-1} \rfloor - \vec{e},$$

if $\left\| \lfloor \gamma\vec{p}R^{-1} \rfloor - \vec{e} \right\| < \gamma/2$, then $[\vec{\mu}]_\gamma = \lfloor \gamma\vec{p}R^{-1} \rfloor - \vec{e}$. Therefore, assuming that $\vec{p}$ can be represented modulo m_σ with no wrap-around, $\vec{p}$ may be obtained as

$$\vec{p} = \vec{c} - \frac{\vec{\mu} - [\vec{\mu}]_\gamma}{\gamma} R \bmod m_\sigma.$$

Algorithm 4 RNS-based Decryption Algorithm for Rose's Cryptosystem

Require:
Require: $d = \det(R) \in \mathbb{Z}$, $\hat{R} = R^{-1}d \in \mathbb{Z}^{n\times n}$, $R' = \gamma\hat{R}B_1 \bmod d$, and $\hat{R}_0 \in \mathbb{Z}^n$ and $R'_0 \in \mathbb{Z}^n$ are the first rows of $\hat{R}$ and R', respectively
Require: $c \in \mathbb{Z}$
Ensure: $\vec{p} \in \mathbb{Z}^n$
1: $\vec{q}_1 = -cR'_0d^{-1} \bmod \mathcal{B}_1$
2: $\vec{q}_2 = \mathrm{FastBConv}(\vec{q}_1, \mathcal{B}_1) \bmod \{\gamma, m_\sigma\}$
3: $\vec{w} = \frac{cR'_0 + \vec{q}_2 d}{B_1} \bmod \{\gamma, m_\sigma\}$
4: $\vec{\mu} = \frac{\gamma c\hat{R}_0 - \vec{w}}{d} \bmod \{\gamma, m_\sigma\}$
5: $\vec{p} = (c, 0, \ldots, 0) - \frac{\vec{\mu} - [\vec{\mu}]_\gamma}{\gamma} R \bmod m_\sigma$
6: **return** $\vec{p}$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

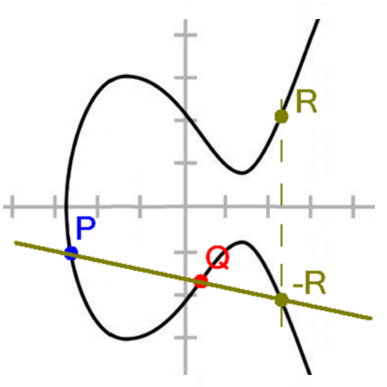


FIGURE 3: The addition R of points P and Q over an elliptic curve E

V. RNS BASED ELLIPTIC CURVE AND ISOGENY CRYPTOGRAPHY

Another operation limiting the performance of emerging cryptography is modular multiplication. There has been extensive research towards making it as efficient as possible [12], [13], [33], [34], [35], since it is featured in ECC [7], [8]. However, one of the most efficient methods for modular arithmetic relies on the selection of finite fields of a specific shape [13] which does not match those of isogeny-based cryptography [17]. A recently proposed number representation [36] that removes this restriction is discussed in this section.

A. ELLIPTIC CURVE AND ISOGENY BASED CRYPTOGRAPHY

ECC corresponds to the design of public-key cryptography based on the algebraic structure of Elliptic Curves (ECs) over finite fields. Points (X, Y) belonging to an EC satisfy (15).

Definition 8 (Elliptic Curve)**.** An elliptic curve E over a field F (denoted $E(F)$) is defined by an equation of the form:

$$Y^2 + aXY + bY = X^3 + cX^2 + dX + e, \qquad (15)$$

where $a, b, c, d, e \in F$.

The structure of ECs might be exploited to define the addition of two of these points. Geometrically, to add $P, Q \in E$, one starts by drawing the line $\overline{PQ}$ (see Fig. 3). Except for the case where $\overline{PQ}$ is vertical, due to the cubic nature of (15), this line will intersect the EC a third time. This point is called $-R$ in the representation in Figure 3 of an EC over $\mathbb{R}$. By reflecting $-R$ across the x axis, the point $R = P + Q$ is produced. The scalar multiplication of $m \in \mathbb{Z}$ by P corresponds to the repeated application of point addition as

$$[m]P = \underbrace{P + \ldots + P}_{m \text{ times}}. \qquad (16)$$

Miller and Koblitz [7], [8] have independently proposed basing the security of cryptosystems on the difficulty of computing the discrete logarithm over elliptic curves defined in finite fields, i.e. of computing m from $[m]P, P \in E(F)$. While these systems reduce the computational complexity of public-key cryptography, when compared to traditional systems like Rivest-Shamir-Adleman (RSA) [2], they are also susceptible to quantum computing attacks [14].

Alternatively, one might define isogenies ϕ over ECs. Isogenies correspond to surjective morphisms with finite kernels, i.e. they map one EC E to another EC E', possibly mapping several points of E to the same point in E', while preserving point addition. Cryptosystems may then be built based on the difficulty of computing ϕ given $P, Q \in E$ and $\phi(P), \phi(Q) \in \phi(E)$, where ϕ has a fixed, smooth, public degree [17] with presumed post-quantum resistance.

The concrete formulae for the computation of point multiplication and isogenies are immaterial for the purpose of this paper. Nonetheless, ECC is defined over fields with prime order or extensions thereof [7], [8], [17]. This enables for a greater flexibility of representations of numbers. While, for instance, the computation of multiplicative roots is made impossible for the kind of rings $\mathbb{Z}_n$ used in RSA, unless one knows the factorisation of n (and hence the corresponding secret-key), this becomes trivial over a finite field of prime order p [37]. This property has recently been exploited to accelerate modular arithmetic over curves with fixed p, currently used for standardised curves [38], [39] and isogeny-based cryptography [17].

B. MODULAR MULTIPLICATION

To accelerate modular multiplications for a generic modulus P, numbers $a \in \mathbb{Z}_P$ are represented in [36] as polynomials $A(X)$ with coefficients of norm smaller than ρ that when evaluated in γ produce:

$$A(\gamma) = a \bmod P, \qquad (17)$$

where γ satisfies $[\gamma^n]_P = \beta$ for a small integer β. Thus, operating with these polynomials modulo $X^n - \beta$ is isomorphic to operating with the corresponding integers modulo P. In [35], it is proven that for digits satisfying $|a^{(i)}| < \rho$, a $\rho \geq \beta P^{1/n}$ suffices to represent all congruency classes $a \in \mathbb{Z}_P$. Digits are represented with respect to two RNS bases $\mathcal{B}_1$ and $\mathcal{B}_2$ and a modulus b_{sk}. The need for these moduli will become evident when describing the Hybrid Polynomial-Residue Number System (HyPoRes) multiplication algorithm.

Definition 9 (HyPoRes)**.** An Hybrid Polynomial-Residue Number System (HyPoRes) is a sextuple $\mathcal{H} = (P, n, \rho, \mathcal{B}_1, \mathcal{B}_2, b_{\mathsf{sk}})$. β is defined to be the smallest integer that is not an n^{th} power over $\mathbb{Z}$, but that has an n^{th} root modulo P. We identify this root with $\gamma^n = \beta \bmod P$. Positive integers $0 \leq a < P$ are represented as a polynomial of n coefficients $(a^{(0)}, \ldots, a^{(n-1)})$, wherein each coefficient $a^{(i)}$ is represented with respect to the two RNS bases $\mathcal{B}_1 = \{b_{1,0}, \ldots, b_{1,h_1-1}\}$ and $\mathcal{B}_2 = \{b_{2,0}, \ldots, b_{2,h_2-1}\}$ and

the modulus b_{sk}, satisfying:

$$a = \sum_{i=0}^{n-1} \left[\sum_{j=0}^{h_1-1} \xi_{i,1,j} \frac{B_1}{b_{1,j}} \right]_{B_1} \gamma^i \bmod P$$

with $\xi_{i,1,j} = \left[a_{i,1,j} \frac{b_{1,j}}{B_j} \right]_{b_{1,j}}$, $|a^{(i)}| < \rho$ and $a_{i,k,j} = a^{(i)} \bmod b_{k,j}$. We use $a_{i,k}$ to denote $a^{(i)} \bmod B_k$, capital values A to denote a representation of a under $\mathcal{H}$, A_1 to denote the representation of a under $\mathcal{B}_1$, A_2 the representation of a under $\mathcal{B}_2$ and a_{sk} the representation of a modulo b_{sk}; and define the norm $||A||_\infty = \max(|a^{(0)}|, \ldots, |a^{(n-1)}|)$.

Algorithm 5 HyPoRes Modular Multiplication Algorithm

Require: $||A||_\infty, ||C||_\infty < k\rho$
Require: $M' = -M^{-1} \bmod \mathcal{B}_1$
Ensure: $||R||_\infty < \rho$ with $r = acB_1^{-1} \bmod P$
$D = A \star C \bmod \mathcal{B}_1 \cup \mathcal{B}_2 \cup \{b_{\mathsf{sk}}\}$
$Q_1 = D \star M' \bmod \mathcal{B}_1$
$Q_2 = \text{FastBConv}(q, \mathcal{B}_1) \bmod \mathcal{B}_2$
$q_{\mathsf{sk}} = \text{FastBConv}(q, \mathcal{B}_1) \bmod b_{\mathsf{sk}}$
$R_2 = \frac{D+Q\star M}{B_1} \bmod \mathcal{B}_2$
$r_{\mathsf{sk}} = \frac{d_{\mathsf{sk}}+q_{\mathsf{sk}}\star M}{B_1} \bmod b_{\mathsf{sk}}$
$\alpha = \left[(\text{FastBConv}(r, \mathcal{B}_2) - r_{\mathsf{sk}}) B_2^{-1} \right]_{b_{\mathsf{sk}}}$
$R_1 = \text{FastBConvSK}(r, \mathcal{B}_2, \alpha) \bmod \mathcal{B}_1$
$R = (R_1, R_2)$

The improved modular multiplication can be found in Algorithm 5. Therein, the operation $A \star C$ denotes the following vector-matrix multiplication:

$$A \star C = AC \bmod (X^n - \beta) =$$
$$\begin{bmatrix} a^{(0)} & a^{(1)} & \ldots & a^{(n-1)} \end{bmatrix} \begin{bmatrix} c^{(0)} & c^{(1)} & \ldots & c^{(n-1)} \\ \beta c^{(n-1)} & c^{(0)} & \ldots & c^{(n-2)} \\ \vdots & \vdots & \ddots & \vdots \\ \beta c^{(1)} & \beta c^{(2)} & \ldots & c^{(0)} \end{bmatrix} \quad (18)$$

where element-wise multiplications are conducted in RNS.

The vector M used in Algorithm 5 corresponds to a small nonzero representation of zero under $\mathcal{H}$. A representation of M with norm smaller than $P^{1/n}$ is guaranteed to exist, as described in Lemma 1.

Lemma 1. *A nonzero representation of zero of norm smaller than $P^{1/n}$ exists under $\mathcal{H}$*

Proof. Let us start by building the lattice $\mathcal{L}(\Gamma)$ of the representations of zero under $\mathcal{H}$ where

$$\Gamma = \begin{bmatrix} P & 0 & \ldots & 0 \\ -\gamma & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -\gamma^n & 0 & \ldots & 1 \end{bmatrix}$$

Each line in Γ corresponds to either $P = 0 \bmod P$ or $-\gamma^i + X^i$, which when evaluated at $X = \gamma$ produces a value congruent with 0. Theorem 2 guarantees that $\mathcal{L}(\Gamma)$ contains a nonzero vector of norm at most $(\det\mathcal{L}(\Gamma))^{1/n} = P^{1/n}$. Thus M can be obtained by finding the nonzero shortest lattice point in $\mathcal{L}(\Gamma)$. While this problem is complex in general, herein we are dealing with lattices of small dimension, typically with $n < 10$, making it solvable in a short time. □

In essence, Algorithm 5 starts by computing $D = A \star C \cong A \times C \bmod P$. Then, to reduce the size of the coefficients, a multiple of a nonzero representation of zero M is added to D, making the result divisible by B_1. Since the scaling factor Q is computed modulo B_1, it is first produced in $\mathcal{B}_1$ and then extended to $\mathcal{B}_2$. Afterwards, $R = \frac{D+Q\star M}{B_1}$ is outputted. The division by B_1 guarantees that the norm of the result is small. However, since this division is not possible in $\mathcal{B}_1$, this value is first produced in $\mathcal{B}_2$ and then extended to $\mathcal{B}_1$.

Algorithm 5 requires that an inverse of M exists in the ring $\mathbb{Z}_{B_1}[X]/(X^n - \beta)$. Lemma 2 guarantees that this is the case when $\mathcal{B}_1$ is built from prime numbers $b_{1,i}$ that do not divide the resultant of M and $X^n - \beta$, and $M \neq 0 \bmod B_1$.

Lemma 2. *When $\mathcal{B}_1 = \{b_{1,0}, \ldots, b_{1,h_1-1}\}$ such that all $b_{1,i}$ are primes not dividing the resultant of M and $X^n - \beta$, and $M \neq 0 \bmod B_1$, M is invertible in $\mathbb{Z}_{B_1}[X]/(X^n - \beta)$.*

Proof. Since $X^n - \beta$ is irreducible, M with $\deg(M) < n$ and $X^n - \beta$ are coprime. Hence, there exists $(U, V) \in \mathbb{Z}[X]^2$ s.t.

$$UM + V(X^n - \beta) = r$$

where r is the resultant of M and $X^n - \beta$, and $r \neq 0$ [40]. We find the inverse of M modulo B_1 and $X^n - \beta$ by computing $Ur^{-1} \bmod b_{1,i}$ for $0 \leq i \leq h_1 - 1$ and lifting the result to $\mathbb{Z}_{B_1}$ with the CRT. The resultant r must be invertible modulo $b_{1,i}$, i.e. coprime to $b_{1,i}$. Since $b_{1,i}$ is prime, it must not divide r. □

Even though values represented under $\mathcal{H}$ will normally satisfy $||A||_\infty < \rho$, their norm might grow after non-reduced additions. Hence, it is assumed that the inputs to Algorithm 5 have their norm bounded by $||A||_\infty, ||C||_\infty < k\rho$. It is also assumed that $\rho < \epsilon B_1$. Notice that since the value of Q is extended from $\mathcal{B}_1$ to $\mathcal{B}_2$ in an inexact way, its norm is bounded by $h_1 B_1$. In this case, the norm of R will satisfy

$$||R||_\infty = \left\| \frac{A \star C + Q \star M}{B_1} \right\|_\infty < \frac{\beta n k^2 \rho^2 + \beta n h_1 B_1 ||M||_\infty}{B_1} < \beta n \epsilon k^2 \rho + \beta n h_1 ||M||_\infty. \quad (19)$$

Since we require that $||R|| < \rho$, ϵ should satisfy:

$$0 < \epsilon < \frac{\rho - \beta n h_1 ||M||_\infty}{\beta n k^2 \rho}$$

As a result, Algorithm 5 produces values with the expected norm as long as $B_1 > \rho/\epsilon$ and $\rho > \beta n h_1 ||M||_\infty$. In addition, FastBConvSK produces the correct value when $\rho < \lambda B_2$ and $b_{\mathsf{sk}} \geq 2(h_2 + \lambda)$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

VI. STOCHASTIC COMPUTING FOR FHE

This section will discuss the results of [41], [42], which enable the methodic design of homomorphic circuits based on stochastic number representations, while efficiently achieving secrecy of the processing algorithm. Rather than trying to provide algorithm-secrecy in general [43]; [41], [42] focus on a subset of applications that can be efficiently evaluated homomorphically. Through the exploitation of stochastic computing, automated mechanisms will be described to map real-valued operations onto the homomorphic domain that result in efficient systems exploiting batching.

A. FULLY HOMOMORPHIC ENCRYPTION

Structured lattices underpin one of the most promising classes of cryptosystems supporting FHE [4]. FHE provides malleable ciphertexts, such that given two ciphertexts encrypting a and b one is able to produce another ciphertext encrypting $a \times b$ or $a + b$. There is noise associated with the ciphertexts that grows as homomorphic operations are applied. Since decryption can only handle a certain amount of noise, there is a limited amount of operations that can be applied before decryption starts associating a message with the wrong value. Gentry [4] has mitigated this problem through bootstrapping, a technique in which ciphertexts are homomorphically decrypted. While the decryption would normally completely remove the noise, homomorphically evaluating the decryption procedure introduces noise. If at the end of bootstrapping the homomorphic capacity of the scheme still allows for one further operation, the process of applying one operation and homomorphically decrypting the result can be indefinitely repeated to achieve FHE.

More modern FHE systems rely on RLWE [44]. With RLWE, techniques have been proposed, which limit the need for bootstrapping, namely by reducing the rate at which noise grows. In [27], modulus-switching is introduced. With this refinement, one can linearly increase the magnitude of the public-key according to the depth of the circuits one wants to process homomorphically, to avoid using bootstrapping. Schemes with this property are said to be leveled FHE schemes.

With the leveled Brakerski-Gentry-Vaikuntanathan (BGV) FHE scheme, the secret-key $\boldsymbol{s} \in \mathcal{R}$ is defined as a "small" polynomial drawn from a distribution χ_{key}. An encryption of $\boldsymbol{m} \in \mathcal{R}_t$, for $t \in \mathbb{N}$, corresponds to a pair of polynomials $\vec{\mathbf{ct}} = (\boldsymbol{c_0}, \boldsymbol{c_1}) \in \mathcal{R}_q^2$ satisfying:

$$[\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q = [[\boldsymbol{m}]_t + t\boldsymbol{v}]_q \tag{20}$$

where $\boldsymbol{v}$ is a noise term that is originally introduced during encryption (which is related to a distribution χ_{err}) and that grows as homomorphic operations are applied.

BGV provides for the homomorphic addition and multiplication of polynomials in $\mathcal{R}_t$. The homomorphic addition of two ciphertexts corresponds to the pairwise addition of the ciphertexts' polynomials. Regarding homomorphic multiplication, it is useful to see ciphertexts as first degree polynomials with coefficients in $\mathcal{R}$. Evaluating a polynomial $\boldsymbol{c_0} + \boldsymbol{c_1 y}$ at $\boldsymbol{y} = \boldsymbol{s}$ leads to (20). In this context, the homomorphic multiplication of $\mathbf{ct}^1$ and $\mathbf{ct}^2$ takes place in two steps. First, $\vec{\mathbf{ct}}_{mult} \leftarrow \left(\left[\boldsymbol{c_0^1 c_0^2}\right]_q, \left[\boldsymbol{c_0^1 c_1^2} + \boldsymbol{c_1^1 c_0^2}\right]_q, \left[\boldsymbol{c_1^1 c_1^2}\right]_q \right)$ is computed. Evaluating $\mathbf{ct}_{mult,0} + \mathbf{ct}_{mult,1}\boldsymbol{y} + \mathbf{ct}_{mult,2}\boldsymbol{y}^2$ at $\boldsymbol{y} = \boldsymbol{s}$ would lead to the decryption of $[\boldsymbol{m}_1 \boldsymbol{m}_2]_t$. Afterwards, one converts the three-element ciphertext back to a two-element ciphertext, through a process called *relinearisation*. In a nutshell, $\mathbf{ct}_{mult,2}$ is multiplied by a pseudo-encryption of $\boldsymbol{s}^2$ and the result is added to $(\mathbf{ct}_{mult,0}, \mathbf{ct}_{mult,1})$. However, since $\mathbf{ct}_{mult,2}$ is uniformly random in $\mathcal{R}_q$, doing this directly would lead to a significant increase of noise after multiplying it by the pseudo-encryption of $\boldsymbol{s}^2$. Two functions are introduced to solve this issue:

$$D_{\omega,q}(\boldsymbol{a}) = \left([\boldsymbol{a}]_\omega, \left[\left\lfloor \boldsymbol{a}\omega^{-1} \right\rfloor\right]_\omega, \ldots, \left[\left\lfloor \boldsymbol{a}\omega^{-(l_{\omega,q}-1)} \right\rfloor\right]_\omega \right) \tag{21}$$

$$P_{\omega,q}(\boldsymbol{a}) = \left([\boldsymbol{a}]_q, [\boldsymbol{a}\omega]_q, \ldots, \left[\boldsymbol{a}\omega^{l_{\omega,q}-1}\right]_q \right) \tag{22}$$

where $l_{\omega,q} = \lfloor \log_\omega(q) + 1 \rfloor$ and $\omega \in \mathbb{N}$.

Instead of multiplying $\mathbf{ct}_{\mathtt{mult},2}$ directly by a pseudo-encryption of $\boldsymbol{s}^2$, $D_{\omega,q}(\mathbf{ct}_{\mathtt{mult},2})$ is multiplied by pseudo-encryptions of $P_{\omega,q}(\boldsymbol{s}^2)$ (designated $\overrightarrow{\mathbf{rlk}}$), and the result is added to the other two elements:

$$\begin{aligned} \vec{\mathbf{ct}}_{relin} \leftarrow \Big(& \left[\mathbf{ct}_{\mathtt{mult},0} + \left\langle D_{\omega,q}(\mathbf{ct}_{\mathtt{mult},2}), \overrightarrow{\mathbf{rlk}}_0 \right\rangle\right]_q, \\ & \left[\mathbf{ct}_{\mathtt{mult},1} + \left\langle D_{\omega,q}(\mathbf{ct}_{\mathtt{mult},2}), \overrightarrow{\mathbf{rlk}}_1 \right\rangle\right]_q \Big) \end{aligned} \tag{23}$$

Since the norm of the elements of $D_{\omega,q}(\mathbf{ct}_{\mathtt{mult},2})$ is at most $\frac{\omega}{2}$, noise growth due to the relinearisation in (23) can be limited by choosing a small ω. Finally, modulus-switching is applied to reduce the growth rate of the norm of $\boldsymbol{v}$ in (20) due to the homomorphic multiplication. This technique consists of scaling the ciphertext to a smaller ring $\mathcal{R}_{q'}$ with an appropriate rounding, which is performed in two steps:

$$\begin{aligned} \boldsymbol{\delta}_i &\leftarrow t \cdot [-\mathbf{ct}_{\mathtt{mult},i}/t]_{q/q'} \text{ for } i = 0, 1 \\ \vec{\mathtt{ct}} &\leftarrow \Big([q'/q \cdot (\mathbf{ct}_{\mathtt{mult},0} + \boldsymbol{\delta}_0)]_{q'}, \\ &\qquad [q'/q \cdot (\mathbf{ct}_{\mathtt{mult},1} + \boldsymbol{\delta}_1)]_{q'} \Big) \end{aligned} \tag{24}$$

As a result of this rounding, noise is similarly scaled down by a factor q'/q.

Batching [5] is a method to improve FHE performance based on the CRT. It allows multiple bits to be encrypted in a single ciphertext, so that one can AND and XOR the bits of sequences of bits using a single homomorphic multiplication or addition. For example, in the considered RLWE cryptosystem, a binary plaintext space has the following structure:

$$\mathcal{P} = \mathbb{Z}[X]/\langle \Phi_m(X), 2 \rangle \tag{25}$$

Φ_m factors modulo 2 into l polynomials $F_0, \ldots, F_{l-1}$ with the same degree d; d is the smallest positive integer satisfying

$2^d = 1(\mathrm{mod} m)$ and $l = \varphi(m)/d$. Through the polynomial CRT, there is a ring isomorphism:

$$\mathcal{P} \cong \mathbb{Z}[X]/\langle F_0(X), 2\rangle \times \ldots \times \mathbb{Z}[X]/\langle F_{l-1}(X), 2\rangle. \quad (26)$$

Batching exploits (26) to encrypt multiple bits in a single ciphertext, so that additions and multiplications operate on them in parallel. To do so, bits $m_0, \ldots, m_{l-1}$ are encoded as a polynomial $m(X)$ satisfying:

$$m_i = m(X) \bmod (F_i(X), 2) \, \forall_{0 \le i < l} \quad (27)$$

Finally, rotations of the plaintext slots can be obtained through mappings of the form $\kappa_i : X \mapsto X^i$. A detailed analysis of these mappings can be found in [45, Section C.3].

Techniques have been proposed in [43] that allow for the offloading of the processing of data, without an homomorphic evaluator knowing the function that is being used for the processing. More concretely, in [43], the emulation of an entire computer architecture is suggested. Since the instructions in memory are also kept encrypted, a homomorphic evaluator has no knowledge of the underlying algorithm. However, [43] only presents experimental results for a toy homomorphic cryptosystem, because homomorphic operations are in general computationally demanding and the cycle by cycle evaluation of a computer architecture is complex. Moreover, since the homomorphic evaluator does not know which instruction is being processed, it is hard to determine when to stop execution.

B. STOCHASTIC COMPUTING BASED FHE

In [41], [42] the "natural" operations that arise from FHE are analysed, and efficient algorithm-hiding systems are designed for applications that take advantage of those operations. Firstly, numbers can be represented as sequences of bits through stochastic representations. Since most FHE schemes support batching as a an acceleration technique for the processing of multiple bits in parallel, one can map stochastic representations to batching slots to efficiently implement the operations in (4) and (5). Unlike traditional Boolean circuits, the amount of bits required for stochastic computing is flexible, thus allowing for a more natural correspondence between the unencrypted and encrypted domains.

The proposed homomorphic systems based on stochastic computing target the approximation of continuous functions. These functions are firstly approximated with Bernstein polynomials, as per (28). Notice that this approximation takes place in a black-box manner. Hence, function development can take place with traditional programming paradigms. Moreover, it also provides for an automated way to produce FHE circuits.

Theorem 2 (Weierstrass Theorem)**.** *If $f : [0,1]^m \to \mathbb{R}$ is a continuous function, then $B_f^{(n_1,\ldots,n_m)}$ converges uniformly to f as $n_1, \ldots, n_m \to \infty$ [46].*

$$\beta_{f,k_1,\ldots,k_m}^{(n_1,\ldots,n_m)} = f\left(\frac{k_1}{n_1}, \ldots, \frac{k_m}{n_m}\right) \quad (28)$$

$$B_f^{(n_1,\ldots,n_m)}(X_1, \ldots, X_m) = \sum_{\substack{0 \le k_l \le n_l \\ l \in \{1,\ldots,m\}}} \beta_{f,k_1,\ldots,k_m}^{(n_1,\ldots,n_m)} \prod_{j=1}^{m} \binom{n_j}{k_j} X_j^{k_j} (1 - X_j)^{n_j - k_j} \quad (29)$$

The coefficients of the Bernstein polynomials resulting from Theorem 2 are encrypted and sent to be homomorphically evaluated. Then, the polynomials are factored as

$$B_f^{(n_1,\ldots,n_m)}(X_1, \ldots, X_m) = \sum_{k_1=0}^{n_1} \binom{n_1}{k_1} X_1^{k_1} (1 - X_1)^{n_1 - k_1} \left(\sum_{k_2=0}^{n_2} \binom{n_2}{k_2} X_2^{k_2} (1 - X_2)^{n_2 - k_2} \cdots \left(\sum_{k_m=0}^{n_m} \beta_{f,k_1,\ldots,k_m}^{(n_1,\ldots,n_m)} \binom{n_m}{k_m} X_m^{k_m} (1 - X_m)^{n_m - k_m} \right) \cdots \right) \quad (30)$$

and Algorithm 1 is recursively called to evaluate them.

This trivially solves the problems faced in [43], albeit at the cost of reducing its range of applicability. An homomorphic evaluator still does not know which function is being evaluated among a wide range of possible continuous functions. Nevertheless, the evaluator knows the degree up to which the function is being approximated, which allows them to know when to terminate computation. Finally, since operations naturally arise from FHE schemes, rather than imposing the structure of a computer architecture on them, this scheme achieves a higher degree of efficiency than [43].

VII. ADDITIVE HOMOMORPHISM FOR DAA

This section considers the representation of values through homomorphisms. This allows for one party to provide their data to another party, while preserving its confidentiality. This property is exploited in [47] to reduce the computational complexity of the DAA. In particular, the additive homomorphism of a commitment scheme is exploited to generate a single proof of knowledge about a secret-key shared between a Trusted Platform Module (TPM) [48] and a Host, instead of two as in traditional schemes [18]. This leads to a twofold reduction of the storage requirements of the TPM, and to a reduction of the signature size of about five times. In this section, we introduce the DAA in the context of TPM and describe the arithmetic aspects of the scheme proposed in [47].

A. DIRECT ANONYMOUS ATTESTATION

DAA is a protocol enabling a trusted platform to authenticate itself to another party in an anonymous way, as represented in Figure 4. The TPM in Figure 4 is a hardware module that can be included, for example, in motherboards, that builds a representation of the state of the Host machine as it boots.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

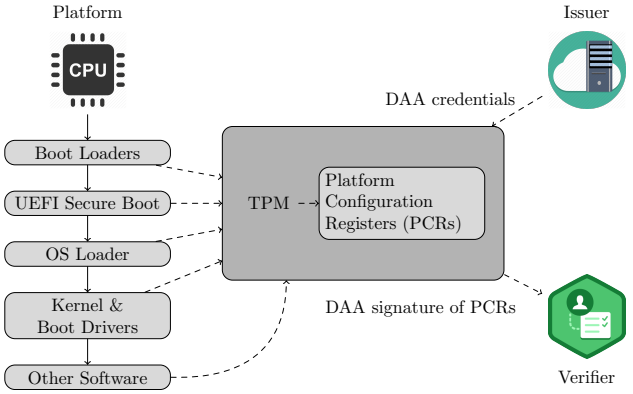


FIGURE 4: TPM-based attestation of a system. As software is loaded, the TPM builds a representation of the platform state. When wishing to prove the trustworthy state of the platform, a DAA signature of the PCRs is produced

Concretely, as software is loaded an hash of it is concatenated with an hash stored in a TPM Platform Configuration Register (PCR), and the result is itself hashed and then stored back in the PCR. Through a DAA signature of the PCRs, the TPM and the Host are able to prove that the platform is in a trustworthy state and that they have been provisioned by an Issuer, and hence that they belong to a certain DAA community.

In a initial phase, TPM-Host pairs are provisioned with DAA credentials by an Issuer. In practice, a DAA credential corresponds to a signature of the signer's identifier produced by the Issuer. Then, when wishing to produce a DAA signature of the PCRs, the TPM and the Host collaborate to create a zero-knowledge proof-of-knowledge of that signature. The zero-knowledge proof-of-knowledge cryptographic construct is used to convince the verifier that the signer possesses a valid membership credential, but without revealing anything else about the identity of the signer. Two mechanisms are put in place to prevent signers from abusing their anonymity. Firstly, should a rogue system make a secret-key publicly available, anyone may check whether a given DAA signature was created under this key or not. Secondly, DAA signatures are created with respect to a basename that is negotiated between the signer and the verifier. If a signer uses the same basename in two signatures, they can be linked, otherwise they cannot.

B. ADDITIVE HOMOMORPHISM AND DAA

The DAA signature includes a zero-knowledge proof-of-knowledge, which is a cryptographic construct used to convince the verifier that the signer possesses a valid membership credential, but without the verifier learning anything else about the identity of the signer. Herein, we describe at a high-level how this proof-of-knowledge may was designed in [47], focusing on how representations through homomorphisms may be leveraged to improve the efficiency of DAA.

In [49], the DAA credential corresponds to a vector of "small" polynomials that is shared between the TPM and the Host. When one computes the inner product between this vector and another one that is a function of the platform identifier, a polynomial made public by the issuer is produced. To prove knowledge of this vector in zero-knowledge, a generalisation of Inhomogeneous Small Integer Solution (ISIS) proofs is used [50]. In [50], to prove the knowledge of a small vector $\vec{x}$ with $||\vec{x}||_\infty \leq \beta$ such that $\vec{x}A = \vec{y} \bmod q$ for a secret $\vec{x} \in \mathbb{Z}_q^n$ and public $A \in \mathbb{Z}_q^{n\times m}$, $\vec{y} \in \mathbb{Z}_q^m$, instead of arguing directly about $\vec{x}$, $\vec{x}$ is decomposed into $k = \lceil \log_2 \beta \rceil$ vectors of norm at most 1:

$$\vec{x} = \sum_{i=1}^{k} 2^{i-1}\vec{b}_i \tag{31}$$

In order to prevent the leakage of the $\vec{b}_i$, elements from $\{-1, 0, 1\}$ are added to the decomposed vectors, so the number of each of them is the same, producing $\vec{x^i} = (\vec{b}_i|\vec{t}_i)$. Finally, the matrix A is also extended with $2n$ 0 rows $A' = (A^T|0^{m\times 2n})^T$ such that:

$$\sum_{i=1}^{k} 2^{i-1}\vec{x^i}A' = \vec{y} \tag{32}$$

The prover now commits to

$$\begin{aligned} c_1 &= \text{COM}\left(\pi_0, \ldots, \pi_{k-1}, \textstyle\sum_{i=1}^{k} 2^{i-1}\vec{r}_i A'\right) \\ c_2 &= \text{COM}\left(\pi_0(\vec{r}_0), \ldots, \pi_{k-1}(\vec{r}_{k-1})\right) \\ c_3 &= \text{COM}\left(\pi_0(\vec{r}_0 + \vec{x^0}), \ldots, \pi_{k-1}(\vec{r}_{k-1} + \overrightarrow{x^{k-1}})\right) \end{aligned} \tag{33}$$

for random $\vec{r}_0, \ldots, \vec{r}_{k-1} \hookleftarrow \mathbb{Z}_q^{3n}$ and uniformly random permutations $\pi_0, \ldots, \pi_{k-1}$. Then, the verifier randomly chooses a challenge $i \hookleftarrow \{1, 2, 3\}$ and the prover reveals $c_j \forall j \neq i$. If c_2, c_3 are revealed, the prover will be convinced that the $\vec{x^i}$ are indeed small. In the other two cases the verifier will be able to validate either the left or the right-hand side of the equation

$$\sum_{i=1}^{k} 2^{i-1}\vec{r}_i A' = \sum_{i=1}^{k} 2^{i-1}(\vec{r}_i + \vec{x^i})A' - \vec{y}, \tag{34}$$

giving the verifier confidence that the prover knows a preimage of $\vec{y}$. Since revealing all commitments would also reveal the $\vec{x}$, the above described process has to be repeated several times.

At a high level two proofs-of-knowledge of this kind are produced in [49]. The TPM and the Host possess $\vec{x}_1$ and $\vec{x}_2$, respectively, such that $\vec{y} = (\vec{x}_1 + \vec{x}_2)A$. To achieve a DAA signature, the TPM produces a signature with respect to $y_1 = (\vec{x}_1 + \vec{t})A$ for a small random $\vec{t}$ that is shared with the Host. Then, the Host produces a proof with respect to $\vec{y}_2 = (\vec{x}_2 - \vec{t})A$. Notice that $\vec{y}_1 + \vec{y}_2 = \vec{y}$, and that $\vec{t}$ is used to randomise the signatures. In [47], in contrast, Baum et al' commitment scheme is exploited since it features additive homomorphism. Based on this property, the TPM and the Host produce commitments that, when homomorphically added,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

result in commitments similar to the ones of (33), but with $\vec{x}_1 + \vec{x}_2$ playing the role of $\vec{x}$ in (31). The TPM and Host may then reveal their replies to the verifier's challenges without anyone learning anything about their secret-key. As a result, a single proof-of-knowledge is required in [47], instead of two as in [49]. As a second optimisation, the commitments are themselves hashed after having been added, resulting in a further reduction in the signature size.

VIII. IMPLEMENTATION AND EXPERIMENTAL RESULTS

This section shows the applicability of the arithmetic, methods and algorithms presented in the previous sections to a wide range of platforms.

A level of parallelism to be considered for massively parallel architectures, such as GPUs, is that of deciphering multiple messages in parallel, by instantiating the same algorithm in multiple work-groups. This type of implementations is particular useful for server settings, where multiple connections are being handled at the same time, and thus several decryptions are required to be performed simultaneously. A similar approach could be applied to multi-core CPU architectures with SIMD extensions. Each SIMD lane would then be used to process the decryption of a different ciphertext in parallel. Nevertheless, as explained in the remainder of this section, the underlying arithmetic provides a great level of parallelism, and thus one can largely accelerate the decryption of a single ciphertext at a time when exploiting parallel architectures, enabling for more flexible parallel systems.

In vector and matrix arithmetic, an inherent level of parallelism relies on the independence of components along the columns. In the case of GGH, the round-off is mainly composed of scalar/vector products. Thus, for instance, the set of products $-cR'_{0,i}d^{-1}$ for $0 \leq i < n$ from Algorithm 4 can be easily scattered on a set of CPU cores or GPU work-items.

The independence of its channels makes the RNS a good match for both Single Instruction stream Multiple Data streams (SIMD) technologies and pipelined architectures. The RNS offers a second level of concurrency (cf. Fig. 5). The basic arithmetic operations are made independently and in a carry-free way on the residues. This can be exploited in different ways, depending on the available resources. For instance, on a multi-core CPU, whereas the first level of parallelization can be supported by multi-threading, the second level can be handled by SIMD instructions sets. Similarly, on a GPU, each work-item can process a single residue without any dependency on other residues. The exploitation of data parallelism is not as viable with multi-precision arithmetic, since the carry propagation chains introduce data and control dependencies, which are not efficiently handled by data parallel architectures, supported for example on SIMD and GPU technologies. To show its wide applicability, herein we describe strategies for the implementation of GGH decryption on highly parallel architectures and of modular multiplication on sequential processors.

For both sequential and parallel implementations, the moduli of the RNS bases were systematically chosen in pseudo-Mersenne form, that is $m = 2^r - c$. The intrinsic modular reduction, inside the rings $\mathbb{Z}/m\mathbb{Z}$, can then be carried out very efficiently through bitwise AND, shifting, and few multiplications with the constant c, as shown in Algorithm 6. This algorithm is based on the following congruency:

$$x = x_2 \times 2^r + x_1 = x_2 \times c + x_1 (\mathrm{mod} m). \quad (35)$$

Algorithm 6 Modular reduction for a pseudo-Mersenne modulus

Require: $m = 2^r - c$, $c < \sqrt{m}$, $\mathtt{mask} = 2^r - 1$, $0 \leqslant x < m^2$

Ensure: $x \bmod m$

1: //loop unrolled based on the worst-case scenario
2: **while** $x \geq 2^r$ **do**
3: $\quad x \leftarrow x \,\&\, \mathtt{mask} + (x \gg r) \times c$
4: **end while**
5: **if** $x \geqslant m$ **then**
6: $\quad x \leftarrow x - m$
7: **end if**
8: **return** x

Two main parameters have a large impact on the performance of schemes supported on cyclotomic rings: the modulus q and the order m of $\Phi_m(X)$. The bitwidth of q defines the homomorphic capacity of the scheme, while a combination of q and m establish the security level. While for power-of-two cyclotomics, the ring arithmetic benefits from efficient Number Theoretic Transforms (NTTs) [51], this choice precludes the exploitation of batching since $\Phi_m(X)$ factors completely modulo 2. Hence, power-of-two cyclomics constitute a good choice for the system described in Section VII but are of limited applicability to the one described in Section VI. Finally, the implementation of homomorphic encryption might benefit from parallelism at multiple levels, for instance pertaining the independence of polynomial coefficients or of the iterations of the i-loop in Algorithm 1.

A. DIVISION AND ROUNDING

In [28], the RNS-based GGH decryption algorithm was implemented and thoroughly tested in three systems, namely i) an Intel i7 4770K CPU with a NVidia K40c GPU, ii) an Intel i7 5960X CPU with a NVidia GTX 980 GPU, and iii) an Intel i7 6700K CPU with a NVidia Titan X GPU. The experimental results, replicated from [28], can be found in Figures 6 and 7. The label "RNS" corresponds to a CPU sequential implementation of the algorithm described in Section IV; "RNS AVX2" to the parallel version of this algorithm, where both multithreading and AVX2 extensions are exploited; "MRS" to the sequential Mixed-Radix System (MRS) algorithm proposed in [52], wherein errors resulting from approximate basis extensions are handled by conversions to a positional system and comparisons; "MRS AVX2"

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

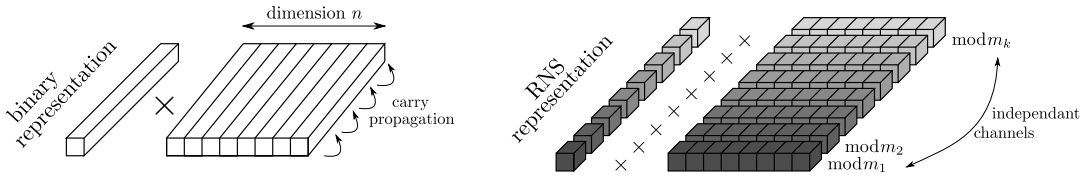


FIGURE 5: Binary and RNS approaches for computing a scalar/vector product

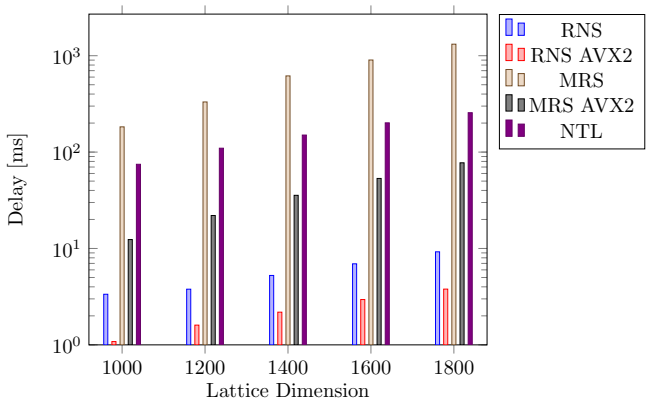


FIGURE 6: Delay of the LBC decryption algorithm for different implementations on the i7 6700K. The y-axis is in logarithmic scale

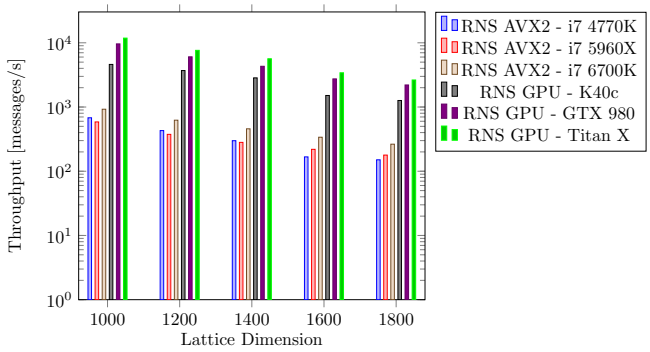


FIGURE 7: Throughput of the RNS AVX2 LBC decryption algorithm on CPU platforms, as well as of the GPU version. The y-axis is in logarithmic scale

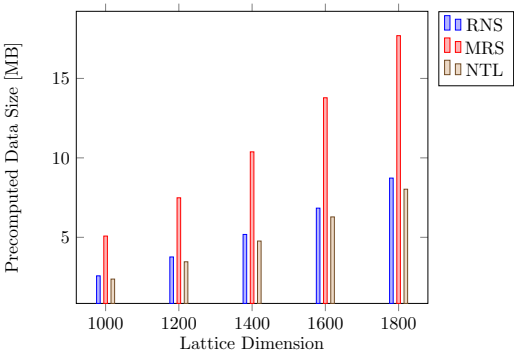


FIGURE 8: Size of the required precomputed data for the LBC decryption algorithm

to the parallel version of the algorithm proposed in [52], where both AVX2 and multithreading are exploited; "NTL" to a multi-precision implementation of decryption using the NTL library [53]; and "RNS GPU" to an RNS implementation on the GPU.

The delay of the MRS implementation proposed in [52] increases rapidly with the dimension of the lattice. Nevertheless, it increases the possibility of exploiting parallelism, which results in a smaller delay for the MRS AVX2 implementation than the NTL algorithm. In contrast, the decryption delay of the RNS and NTL approaches increases at a slow pace. Furthermore, the RNS approach provides greater opportunities to benefit from parallelism, which makes the RNS-based multithreaded AVX2 algorithm the most efficient with respect to the delay. The throughput of the different implementations is represented in Fig. 7. We can see the sustained throughputs of the RNS GPU implementations are much higher than those of the CPU. This derives from the fact that the algorithm proposed in [28] requires a significant less amount of synchronisations and memory accesses than what would be necessary for [52], due to the avoidance of the computation of the MRS digits. A maximum throughput of 11832 messages/s is obtained for Titan X and for the dimension 1000.

The typical size of the precomputed data required for the RNS, MRS and NTL versions of the decryption algorithm can be found in Fig. 8. We can see that the amount of extra data required by the RNS approach when compared with the NTL is negligible, with an average increase in size of 8.7%. Moreover, one can see that this size is more than halved for the RNS approach when compared with the MRS, since one no longer needs a second large base, and neither the precomputed data associated with it, nor the data associated with computing the mixed-radix digits.

B. MODULAR MULTIPLICATION

The HyPoRes method for modular multiplication was implemented and experimentally evaluated in [36], along with [13], [11], for comparison. The pure RNS-based multiplication [11] can be seen as a simplification of [36] when $n = 1$, and thus $M = P$ and γ and β play no role. The Hybrid-Positional Residue Number System (HPR) representation of [13] makes use of a positional system wherein the digits are represented in RNS, but requires the selection of primes of the form $P = B_1^n - \beta$ to achieve efficient modular reduc-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

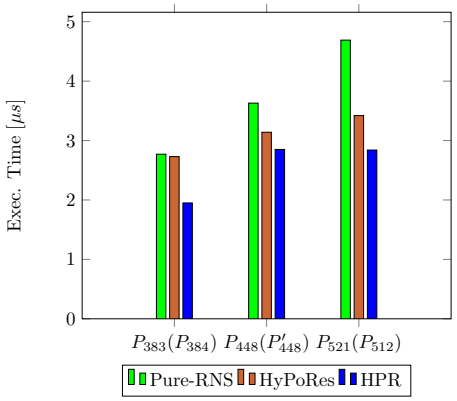


FIGURE 9: Average execution time of modular multiplication following a pure-RNS and [36], as well as of with HPR with specially crafted primes. The settings in parenthesis were used for the HPR modular multiplication

System	Encryption [s] Intel / Arm	Filter [s] Intel	Decryption [s] Intel / Arm
Grey Stretching Fixed-point	52.5 / 685	341	6.9 / 134
Blending Fixed-point	52.7 / 684	885	5.3 / 88
Grey Stretching Stochastic	34.5 / 914	1340	61.7 / 1172
Blending Stochastic	47.7 / 1273	2103	89.4 / 1468

TABLE 2: Average execution time for homomorphic image processing operations on an Intel i7-5960X CPU and on a Arm Cortex-A53 CPU

tions. After computing a product, the Most Significant Digits (MSDs) are scaled by β and added to the Least Significant Digits (LSDs). Then, carry propagation is applied through a mechanism involving basis extensions. Experimental results in [36] were obtained for the primes P_{383}, P_{448} and P_{521} used to define the elliptic curves M-383, Ed448-Goldilocks and E-521 [38], [39], respectively. Moreover, HPR-crafted primes P_{384}, P'_{448} and P_{512} of 384, 448 and 512 bits, respectively, have been considered for the implementation of [13].

The average modular multiplication times for the HyPoRes, pure-RNS and HPR representations from [36] have been replicated in Fig. 9. Fig. 9 suggests that, although a similar performance is attained for both HyPoRes and a pure-RNS approach for the prime P_{383}, the HyPoRes system has a better scalability as the bit-length of the primes increases. Moreover, a maximum speed-up of approximately 1.4 is obtained for P_{521}. While Fig. 9 shows that the performance of HyPoRes is slightly worse than HPR, it relies on weaker assumptions (since it does not require the use of specially crafted primes), making it more flexible and applicable in practice. In particular, since HPR relies on primes of a particular kind, this makes it hardly practical, because the primes for cryptographic applications have already been standardised with a different shape. In contrast, HyPoRes can be used whenever one knows the factoring of the underlying modulus. While the applicability to ECC has been herein demonstrated, HyPoRes can also be applied to ElGamal [3] and RSA [2] decryption and signing.

C. STOCHASTIC COMPUTING FOR FHE

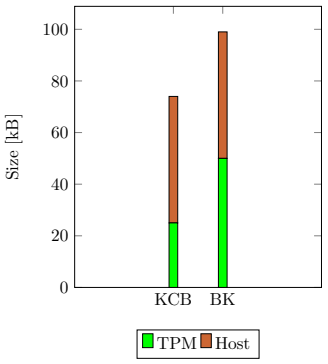
The method described in Section II-B, which exploits stochastic computing for efficient algorithm-hiding FHE, was implemented in [41]. Therein, it was also considered the conversion of the Bernstein polynomials that result from Theorem 2 to power form, along with their evaluation supported on a fixed-point representation approach proposed in [54]. It should be noted that the stochastic representation proposed in [41] is more widely applicable, since most FHE schemes support batching, while [54] is only applicable to BGV. Two piece-wise multilinear functions were developed for image processing, namely grey stretching and image blending, and their encrypted approximations were automatically derived using the methods described in Section VI. Functions (36) and (37), corresponding to grey stretching and image blending, respectively, were applied to images with 256×256 pixels from [55] and pixels were mapped to the $[0, 1]$ interval. The fixed-point approach of [54] was implemented with the NFLlib [51], and exploited a cyclotomic polynomial Φ_m with $m = 2^{14}$, and a modulus with $\log_2 q \approx 372$. For the stochastic number representation, HElib [56] was used with $m = 4369$ and $m = 5461$ (accounting for 256 and 378 batching slots), and moduli $\log_2 q \approx 132$ and $\log_2 q \approx 324$ were applied for the grey stretching and blending algorithms, respectively.

$$g(x_1) = \begin{cases} 0, & \text{if } x_1 \leq 0.25 \\ 2x_1 - 0.5, & \text{if } 0.25 < x_1 \leq 0.75 \\ 1, & \text{otherwise.} \end{cases} \tag{36}$$

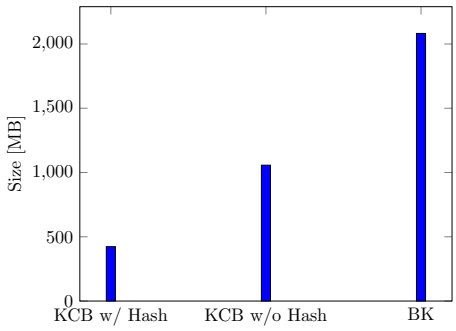
$$b(x_1, x_2) = \begin{cases} x_1 x_2, & \text{if } x_1 \leq 0.5 \\ 1 - 2(1 - x_1)(1 - x_2), & \text{otherwise.} \end{cases} \tag{37}$$

The average execution times of encrypting and decrypting images on a high-performance Intel i7-5960X CPU and on an embedded Arm Cortex-A53 CPU and homomorphically processing on the i7-5960X featured in [41] can be found in Table 2. The timings of encrypting and decrypting images on the quad-core Cortex-A53 have been included to have experimental results representative of scenarios where data may be encrypted or decrypted on an embedded device and processed in the cloud. Since the considered homomorphic image processing algorithms show a great level of parallelism, 16 threads were created on the i7-5960X processor and 4 threads on the Cortex-A53. Each thread sequentially

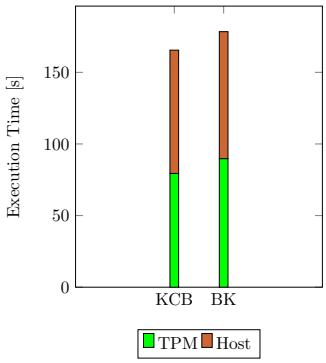
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access



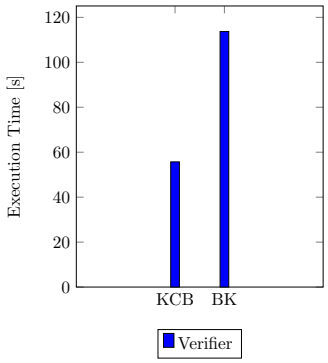
(a) Private-key material for [47] (KCB) and [18] (BK)



(b) Signature size for [47] scheme with hashed commitments (KCB w/ Hash), with regular commitments (KCB w/o Hash) and [18] (BK)



(c) Signing performance for [47] (KCB) and [18] (BK)



(d) Verification performance for [47] and [18]

FIGURE 10: Experimental results comparing the DAA protocol proposed in [47] and [18]

processed different parts of the image. Due to the limited resources of the embedded device, encryption and decryption are, on average, 19.8 and 17.9 times slower to execute on the Arm than on the Intel processor, respectively. Nevertheless, acceptable performance, in the order of hundreds of seconds, is still achieved on the Arm processor.

Fixed-point homomorphic arithmetic made use of efficient power-of-two cyclotomic rings, while stochastic representations made use of non-power-of-two cyclotomics. This is reflected in Table 2. For the two considered processors, encryption takes the same time for both grey stretching and blending when using a fixed-point approach, because both use the same parameter set. With a stochastic number representation, the use of non-power-of-two cyclotomics allows for the use of smaller parameters for grey stretching than for blending, leading to a more efficient encryption for the former filter. Nevertheless, as homomorphic operations contribute to the reduction of the bit-size of the ciphertext modulus, decrypting the result on both processors after blending images takes a shorter period than after grey stretching for the fixed-point approach since the former application is more complex. Finally, while the fixed-point approach is less applicable than stochastic computing, since more FHE schemes support batching than rescaling, the fixed-point approach takes from 2.4 up to 3.9 times less time to process images.

D. ADDITIVE HOMOMORPHISM FOR DAA

In order to evaluate the efficiency of homomorphic representations, both the scheme described in Section VII and [18] were implemented in [47]. The implementations made use of the following cryptographic parameters: $n = 256$, $q = 8380417$, $l = 32$, $m = 24$ and $\beta = 256$. These parameters are only used for comparative purposes. All entities, namely the TPM, the Issuer, the Host and the Verifier, were simulated on an Intel i9 7900X CPU with 64GB running at 3.3 GHz operated by CentOS 7.5. We replicated the experimental results obtained in [47] herein. Fig. 10a shows that in practice the size of the TPM private-key share is halved when [47] is compared with [18]. This is of particular importance for TPM platforms, where memory resources are constrained. In Fig. 10b, signature sizes are shown for [47], both when commitments are hashed and when they are not, and for [18]. The signature size of [47] is halved, when hashing is not considered, in comparison to the scheme in [18]. This improvement is achieved through the exploitation of Baum et al's commitment scheme, enabling the construction of a single proof of knowledge that reflects the secret-key shared between the TPM and Host, instead of the two required by [18]. Moreover, by hashing commitments, the size of the proofs are themselves reduced, resulting in signatures that are 5 times smaller than [18].

Fig. 10c shows that the signing operation of [47] is faster than [18]. More noticeably, the verification operation is significantly enhanced when comparing [47] to [18], as shown in Fig. 10d. Since a single proof needs to be verified instead of two, a speed-up of more than 2 is achieved. Notice that in Figures 10c and 10d the execution times include the hashing of the commitments.

IX. CONCLUSIONS AND FUTURE RESEARCH

This paper highlights the importance of investigating alternative number representations, like the RNS, stochastic representations and homomorphisms, to achieve efficient, adaptable and secure emerging cryptosystems. While straightforward implementations of emerging cryptography would make use of binary representations, they would lead to long carry chains, precluding the exploitation of data parallelism, and multiplication algorithms of quadratic complexity. Also, FHE provides clients with unconventional instructions, onto which the mapping of binary representations might be sub-optimal. Moreover, homomorphisms allow for the offloading of parts of a computation without breaking the confidentiality of the associated data, and might be used to reduce the workload of embedded devices processing emerging cryptographic primitives.

The methods, algorithms and techniques herein surveyed show that the beneficial aspects of the application of the RNS to traditional cryptography are portable to post-quantum cryptography. The interest of RNS systems has been generalised and confirmed for emerging cryptography, since: i) they do not incur on the overhead of general-purposed multi-precision arithmetic libraries; ii) they make exclusive use of integer operations and hence naturally benefit from architectural improvements to computer arithmetic units; and iii) they reduce the asymptotic complexity of isogeny-based and GGH-like cryptosystems, potentiating the diversification of security assumptions in a post-quantum world. Moreover, there has been a long stream of research on the application of the RNS to FHE, but this has been excluded from this paper for the sake of showing further examples of unorthodox number representations applicable to cryptography.

FHE offers a set of instructions onto which users must map their applications, that is radically different from traditional programming. While one could force the structure of binary representations onto FHE, the resulting systems would be sub-optimal. It has been shown that stochastic representations enable not only a natural way for users to map their applications onto the homomorphic domain, but also that stochastic operations have direct homomorphic counterparts, leading to low-depth circuits for scaled addition and multiplication. Moreover, by encrypting the coefficients of the polynomials that approximate homomorphic functions, one achieves the confidentiality of the processing algorithm in an efficient way.

The interpretation described in this paper of homomorphisms as a representation of data that protects their confidentiality has led to the acceleration of a lattice-based DAA protocol. As a byproduct, the size of the key-material was significantly reduced. This is particularly important to bring developments of quantum-resistant cryptography onto the realm of practicality for devices with restricted computing resources, like the TPM.

All in all, experimental results confirm the interest of this survey on using non-positional arithmetic for emerging cryptographic, not only for current processing platforms but also for the future highly parallel and heterogeneous systems and embedded systems with severe computing restrictions.

Cryptography is a multidisciplinary topic, and one cannot ignore the potential impact that the number representation techniques highlighted in this survey may have on the conception of future cryptographic systems. These systems may be designed from the start targeting a specific number representation, making them more computationally efficient on the targeted platforms. Moreover, while this survey has mostly focused on metrics such as execution time and memory consumption, future research should evaluate also the impact of number representations on other aspects, including energy consumption and resistance against side-channel attacks. The main risk of moving away from traditional number representations has to do with a lack of tools, and the inability to leverage the knowledge built for the implementation of traditional cryptography. Nevertheless, the methods and experimental results presented in this paper give us confidence that the long term benefits of this novel approach largely outweigh the risks. Finally, a panoply of other cryptographic systems may be targeted in the future, with applications ranging from cryptocurrencies to attribute-based encryption.

REFERENCES

[1] Alfred J Menezes, Scott A Vanstone, and Paul C Van Oorschot. Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[2] R L Rivest, A Shamir, and L Adleman. A Method for Obtaining Digital Signatures and Public-key Cryptosystems. Commun. ACM, 21(2):120–126, 1978.

[3] Taher El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In Proceedings of CRYPTO 84 on Advances in Cryptology, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[4] Craig Gentry. Fully Homomorphic Encryption Using Ideal Lattices. In Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.

[5] N P Smart and F Vercauteren. Fully Homomorphic SIMD Operations. Cryptology ePrint Archive, Report 2011/133, 2011.

[6] Yin Hu. Improving the Efficiency of Homomorphic Encryption Schemes. PhD thesis, Worcester Polytechnic Institute, 5 2013. https://web.wpi.edu/Pubs/ETD/Available/etd-042513-154859/unrestricted/YHu.pdf.

[7] Victor S Miller. Use of elliptic curves in cryptography. In Hugh Williams, editor, Lecture Notes in Computer Science: Advances in Cryptology - CRYPTO 1985 Proceedings, pages 417–426. Springer Berlin / Heidelberg, Berlin, Heidelberg, 1985.

[8] Neal Koblitz. Elliptic curve cryptosystems. Mathematics of Computation, 48(177):203–209, January 1987.

[9] Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, and Vashek Matyas. The return of Coppersmith's attack: Practical factorization of widely used RSA moduli. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17, pages 1631–1648, New York, NY, USA, 2017. ACM.

[10] D J Bernstein, J Buchmann, and E Dahmen. Post Quantum Cryptography. Springer Publishing Company, Incorporated, 1st edition, 2008.

[11] Samuel Antao, Jean Claude Bajard, and Leonel Sousa. RNS-based elliptic curve point multiplication for massive parallel architectures. The Computer Journal, 55:629–647, 05 2012.

[12] Karim Bigou and Arnaud Tisserand. Single Base Modular Multiplication for Efficient Hardware RNS Implementations of ECC. pages 123–140. Springer, Berlin, Heidelberg, 2015.

[13] K. Bigou and A. Tisserand. Hybrid position-residues number system. In 2016 IEEE 23nd Symposium on Computer Arithmetic (ARITH), pages 126–133, July 2016.

[14] P W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, pages 124–134, Nov 1994.

[15] NIST. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. Technical report, NIST, 2016.

[16] Chris Peikert. A Decade of Lattice Cryptography. Found. Trends Theor. Comput. Sci., 10(4):283–424, 2016.

[17] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J. Mathematical Cryptology, 8(3):209–247, 2014.

[18] Rachid El Bansarkhani and Ali El Kaafarani. Direct anonymous attestation from lattices. Cryptology ePrint Archive, Report 2017/1022, 2017. https://eprint.iacr.org/2017/1022.

[19] Peter L. Montgomery. Modular multiplication without trial division. Mathematics of Computation, 44(170):519–519, may 1985.

[20] Jean-Claude Bajard and Laurent Imbert. A full RNS implementation of rsa. IEEE Trans. Comput., 53(6):769–774, June 2004.

[21] Jean-Claude Bajard, Julien Eynard, Anwar Hasan, and Vincent Zucca. A full RNS variant of FV like somewhat homomorphic encryption schemes. Cryptology ePrint Archive, Report 2016/510, 2016. https://eprint.iacr.org/2016/510.

[22] B R Gaines. Stochastic Computing Systems, pages 37–172. Springer US, Boston, MA, 1969.

[23] Zbyněk Šír and Bert Juttler. On de Casteljau-type algorithms for rational Bézier curves. Journal of Computational and Applied Mathematics, 288:244–250, 2015.

[24] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. Efficient commitments and zero-knowledge protocols from ring-sis with applications to lattice-based threshold cryptosystems. Cryptology ePrint Archive, Report 2016/997, 2016. https://eprint.iacr.org/2016/997.

[25] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S Kaliski, editor, Proceedings of Advances in Cryptology, volume 1294 of LNCS, pages 112–131. Springer Berlin Heidelberg, 1997.

[26] Ravi Kannan. Minkowski's convex body theorem and integer programming. Mathematics of Operations Research, 12(3):377–568, 8 1987.

[27] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) Fully Homomorphic Encryption Without Bootstrapping. ACM Trans. Comput. Theory, 6(3):13:1—-13:36, 2014.

[28] Paulo Martins, Julien Eynard, Jean-Claude Bajard, and Leonel Sousa. Arithmetical Improvement of the Round-Off for Cryptosystems in High-Dimensional Lattices. IEEE Transactions on Computers, 66(12):2005–2018, 2017.

[29] P Martins, L Sousa, J Eynard, and J.-C. Bajard. Programmable RNS lattice-based parallel cryptographic decryption. In Proceedings of 26th International Conference on Application-specific Systems, Architectures and Processors (ASAP), pages 149–153, 2015.

[30] Jean-Claude Bajard, Julien Eynard, Nabil Merkiche, and Thomas Plantard. RNS arithmetic approach in lattice-based cryptography: Accelerating the "rounding-off" core procedure. In 22nd IEEE Symposium on Computer Arithmetic, ARITH 2015, Lyon, France, June 22-24, 2015, pages 113–120. IEEE, 2015.

[31] Michael Rose, Thomas Plantard, and Willy Susilo. Improving BDD Cryptosystems in General Lattices. In Feng Bao and Jian Weng, editors, Proceedings of Information Security Practice and Experience, volume 6672 of LNCS, pages 152–167. Springer Berlin Heidelberg, 2011.

[32] L Babai. On Lovász' Lattice Reduction and the Nearest Lattice Point Problem (Shortened Version). In Proc. of the 2Nd Symp. of Theoretical Aspects of Comput. Sci., STACS '85, pages 13–20, London, UK, 1985.

[33] Samuel Antão and Leonel Sousa. The CRNS framework and its application to programmable and reconfigurable cryptography. ACM Transactions on Architecture and Code Optimization, 9(4):1–25, jan 2013.

[34] Samuel Antão, Jean-Claude Bajard, and Leonel Sousa. RNS-based elliptic curve point multiplication for massive parallel architectures. The Computer Journal, 55(5):629–647, 2011.

[35] J.-C. Bajard, L. Imbert, and T. Plantard. Arithmetic operations in the polynomial modular number system. In 17th IEEE Symposium on Computer Arithmetic (ARITH'05), pages 206–213, June 2005.

[36] Paulo Martins, Jérémy Marrez, Jean-Claude Bajard, and Leonel Sousa. Hypores: An hybrid representation system for ECC. In IEEE 26th Symposium on Computer Arithmetic (ARITH), June 2019.

[37] Anna M. Johnston. A generalized qth root algorithm. In Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '99, pages 929–930, Philadelphia, PA, USA, 1999. Society for Industrial and Applied Mathematics.

[38] Diego F. Aranha, Paulo S. L. M. Barreto, Geovandro C. C. F. Pereira, and Jefferson E. Ricardini. A note on high-security general-purpose elliptic curves. Cryptology ePrint Archive, Report 2013/647, 2013. https://eprint.iacr.org/2013/647.

[39] Mike Hamburg. Ed448-goldilocks, a new elliptic curve. Cryptology ePrint Archive, Report 2015/625, 2015. https://eprint.iacr.org/2015/625.

[40] Joe Buhler. Resultants, discriminants, bezout, nullstellensatz, etc. http://people.reed.edu/~jpb/alg/notes/101.pdf, 2001. Reed College.

[41] Paulo Martins and Leonel Sousa. A methodical FHE-based cloud computing model. Future Generation Computer Systems, 95:639 – 648, 2019.

[42] Paulo Martins and Leonel Sousa. A Stochastic Number Representation for Fully Homomorphic Cryptography. In 2017 IEEE International Workshop on Signal Processing Systems (SiPS), pages 1–6. IEEE, oct 2017.

[43] Michael Brenner, Jan Wiebelitz, Gabriele von Voigt, and Matthew Smith. Secret program execution in the cloud applying homomorphic encryption. In 5th IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2011), pages 114–119. IEEE, may 2011.

[44] Zvika Brakerski and Vinod Vaikuntanathan. Fully Homomorphic Encryption from ring-LWE and Security for Key Dependent Messages. In Proceedings of the 31st Annual Conference on Advances in Cryptology, CRYPTO'11, pages 505–524, Berlin, Heidelberg, 2011. Springer-Verlag.

[45] Craig Gentry, Shai Halevi, and Nigel P Smart. Fully Homomorphic Encryption with Polylog Overhead, pages 465–482. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[46] C Heitzinger. Simulation and Inverse Modeling of Semiconductor Manufacturing Processes. 2002.

[47] Nada El Kassem, Liqun Chen, Rachid El Bansarkhani, Ali El Kaafarani, Jan Camenisch, Patrick Hough, Paulo Martins, and Leonel Sousa. More efficient, provably-secure direct anonymous attestation from lattices. Future Generation Computer Systems, 99:425 – 458, 2019.

[48] Will Arthur and David Challener. A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security. Apress, Berkely, CA, USA, 1st edition, 2015.

[49] Rachid El Bansarkhani and Ali El Kaafarani. Direct anonymous attestation from lattices. Cryptology ePrint Archive, Report 2017/1022, 2017. https://eprint.iacr.org/2017/1022.

[50] San Ling, Khoa Nguyen, Damien Stehlé, and Huaxiong Wang. Improved zero-knowledge proofs of knowledge for the isis problem, and applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, Public-Key Cryptography – PKC 2013, pages 107–124, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[51] Carlos Aguilar-Melchor, Joris Barrier, Serge Guelton, Adrien Guinet, Marc-Olivier Killijian, and Tancrede Lepoint. NFLlib: NTT-based Fast Lattice Library. In RSA Conference Cryptographers' Track, San Francisco, United States, 2016.

[52] J.-C. Bajard, J Eynard, N Merkiche, and T Plantard. RNS Arithmetic Approach in Lattice-Based Cryptography: Accelerating the "Rounding-off" Core Procedure. In Proceedings of 22nd Symposium on Computer Arithmetic, pages 113–120, 2015.

[53] Victor Shoup. NTL 9.11.0: A library for doing number theory. www.shoup.net/ntl, 2016.

[54] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic Encryption for Arithmetic of Approximate Numbers. Cryptology ePrint Archive, Report 2016/421, 2016.

[55] University of Granada. Test Images. http://decsai.ugr.es/cvg/dbimagenes/, 2014.

[56] Shai Halevi and Victor Shoup. Algorithms in HElib, pages 554–571. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983020, IEEE Access

PAULO MARTINS received a Ph.D. degree in Electrical and Computer Engineering from the Instituto Superior Técnico (IST), Universidade de Lisboa (UL), Lisbon, Portugal, in 2019. He was a junior researcher in the R&D Instituto de Engenharia de Sistemas e Computadores (INESCID) and is currently at Samsung Research United Kingdom. He did a collaboration with the Laboratoire d'informatique de Paris 6 (LIP6) during 4 months in 2016. His research interests include cryptography, computer architectures, parallel computing, and computer arithmetic. He is also a member of both IEEE and HiPEAC.

LEONEL SOUSA received a Ph.D. degree in Electrical and Computer Engineering from the Instituto Superior Técnico (IST), Universidade de Lisboa (UL), Lisbon, Portugal, in 1996, where he is currently Full Professor. He is also a Senior Researcher with the R&D Instituto de Engenharia de Sistemas e Computadores (INESC-ID). His research interests include computer arithmetic, VLSI architectures, parallel computing and signal processing. He has contributed to more than 200 papers in journals and international conferences, for which he got several awards, including: DASIP'13 Best Paper Award, SAMOS'11 'Stamatis Vassiliadis' Best Paper Award, DASIP'10 Best Poster Award, and the Honorable Mention Award UTL/Santander Totta for the quality of the publications in 2009 and 2016. He has contributed to the organization of several international conferences, namely as program chair and as general and topic chair, and has given keynotes in some of them, he is member of the Program Committee of the IEEE International Symposium on Computer Arithmetic in 2020. He has edited four special issues of international journals, he is currently Associate Editor of the IEEE Transactions on Computers, IEEE Access and Springer JRTIP, and Senior Editor of the Journal on Emerging and Selected Topics in Circuits and Systems. He has co-edited the books Circuits and Systems for Security and Privacy (CRC, 2018) and Embedded Systems Design with Special Arithmetic and Number Systems (Springer, 2017). He is Fellow of the IET, Distinguished Scientist of ACM and Senior Member of IEEE.