Abstract Homomorphic Encryption is a recent promising tool in modern cryptography, that allows to carry out operations on encrypted data. In this paper we focus on the design of a scheme based on pairings and elliptic curves, that is able to handle applications where the number of multiplication is not too high, with interesting practical efficiency when compared to lattice based solutions. The starting point is the Boneh-Goh--Nissim (BGN for short) encryption scheme [1], which enables the homomorphic evaluation of polynomials of degree at most 2 on ciphertexts. In our scheme, we use constructions coming from [2,3], to propose a variant of BGN scheme that can handle the homomorphic evaluation of polynomials of degree at most 4. We discuss both the mathematical structure of the scheme, and its implementation. We provide simulation results, showing the relevance of this solution for applications requiring a low multiplicative depth, and give relative comparison with respect to lattice based homomorphic encryption schemes.

Keywords Homomorphic encryption, pairing-based cryptography, elliptic curves, low depth circuits.

Design and Implementation of Low Depth Pairing-based Homomorphic Encryption Scheme

Vincent Herbert · Bhaskar Biswas · Caroline Fontaine

1 Introduction

Motivations. *Homomorphic Encryption* is a recent promising tool in modern cryptography, as it allows to carry out operations on encrypted data. The key idea is that performing some operations on encrypted data provide the same result after decryption as if the computation would have been performed on the original plain data. Furthermore, with such a tool one could outsource storage and/or computation without endangering data's privacy. While some well established encryption schemes sometimes offer homomorphic properties, for addition [4] or multiplication [5] operations, they do not provide a way to perform both additions and multiplications at the same time. A few recent schemes have been proposed that can handle more.

On one hand, there are a few number of schemes which are based on well-established security assumptions like Discrete Logarithm computation [6],these schemes being able to handle an arbitrary number of additions and a few number of multiplications. Pairing-based cryptography [7] offers the Boneh-Goh-Nissim scheme [1], which is able to natively handle any number of additions and one single multiplication. Moreover, with the Catalano-Fiore
construction [3], it is possible to improve some existing schemes to help them manage one more multiplication depth. Hence, for example, whereas the well known Paillier scheme [4] can only handle additions, it is possible to modify it with the Catalano-Fiore result to get a scheme that is able to handle not only additions but also one multiplication.

On the other hand there exist several families of schemes built upon lattices [8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26] or coding theory [27]. These schemes are able to handle in theory an arbitrary number of additions and a larger (or even unlimited) number of multiplications. Unfortunately, most of these latter schemes lead to expensive running times and/or ciphertext sizes, and even if progresses are impressive they are probably oversized for applications that only need a few number of multiplications. Moreover, these schemes' security is based on security assumptions — such as SPRP [28,29,30], LWE [31], RLWE [32], etc. — that are not sufficiently well mastered at the moment to provide robust setting guidelines.

Hence, if we focus on use cases that do not need too many multiplications to be chained, *e.g.* for Pearson test computation like in [12], it is always of interest to closely look at lighter and more secure solutions. This motivated us to work in this paper on the design of a new scheme based on long-studied security assumptions, which can handle more than one multiplication.

Contributions. Our scheme is based on an improvement of [1], that we call BGN-F-CF, and which can homomorphically evaluate polynomial of degree at most 4. This solution provides smaller ciphertexts than lattice based solutions, and its security is based on a hard problem that has been investigated in depth. More precisely, it employs together two improvements of the original BGN scheme [1], based on [2] and [3]. Only Freeman's work [2] has already been coupled with BGN in [33,34] to greatly improve BGN's speed, resulting in a scheme that we call here BGN-F. To our knowledge, it is the first time that Catalone and Fiore's construction [3] is

Vincent Herbert
CEA Paris-Saclay, France (This work was done while the author was at CNRS, Lab-STICC).

Bhaskar Biswas
CNRS Lab-STICC and IMT-Atlantique, Brest, France.

Caroline Fontaine
CNRS Lab-STICC and IMT-Atlantique, Brest, France.

applied to this particular setting in order to add one more multiplicative depth. In this paper, we discuss both the mathematical structure of the scheme, and its implementation.

Organization of the paper. This paper is organized as follows. We first recall in Section 2 some useful information about the original BGN scheme, and provide a high-level description of its variants BGN-F and BGN-F-CF. Then, we describe our scheme BGN-F-CF in depth in Sections 3 and 4. While Section 3 provides a high-level description of the scheme, we give in Section 4 more details about the instanciation we used for our implementation. The security is discussed in Section 5. The performances and simulation results are presented in Section 6. We also provide some comparison with lattice based solutions, showing the interest of BGN-F-CF for the homomorphic evaluation of small degree polynomials. Section 7 finally concludes the paper. We present some related additional materials in appendix. In particular, we describe some elements concerning ciphertext constraints for different low multiplicative depth circuits and provide more precise details of parameter settings used in the experiments.

2 Preliminaries

2.1 Related works

Our scheme BGN-F-CF is a variant of the Boneh-Goh-Nissim (BGN for short) scheme [1]. BGN is a homomorphic public-key pairing-based encryption scheme that allows the evaluation of a multivariate quadratic polynomial over small encrypted integers. This cryptosystem brings into play one composite-order cyclic group G and is semantically secure [3, page 7] under the subgroup decision assumption over G [1, page 3]. Instantiations use a symmetric pairing (*i.e.* the non-degenerate bilinear map is symmetric) and the group of points on a super-singular (i.e. $\#E(\mathbb{F}_p) = p + 1$) elliptic curve E defined over a prime field $\mathbb{F}_p$. The group order must be impossible to factor. This leads to manage a big-order group for standard security level and, thus, slowing pairing computation. A solution was proposed by Freeman in [2] to convert pairing-based cryptosystems from composite-order groups to prime-order groups. It can particularly be applied on BGN to obtain a variant, that we call BGN-F, with two prime-order cyclic groups $\mathbb{G}$ and $\mathbb{H}$. Instantiations use an asymmetric pairing and two direct groups $G = \mathbb{G}^2$ and $H = \mathbb{H}^2$ where $\mathbb{G}$ and $\mathbb{H}$ are two subgroups of a pairing-friendly [35] ordinary elliptic curve E defined over $\mathbb{F}_{p^k}$[1]. BGN-F is semantically secure under the(generalized) subgroup decision assumption in G and H [2, page 49] or equivalently under the decisional Diffie-Hellman (DDH) assumption in $\mathbb{G}$ and $\mathbb{H}$ [2, page 58]. This implies the security in the target group $\mathbb{G}_T$ of the pairing under the elliptic curve Diffie Hellman (ECDH) assumption. Both schemes are compared in [33,34]. In 2015, Catalano and Fiore proposed in [3] a technique to allow one more homomorphic multiplication that the underlying homomorphic encryption scheme can support. This technique needs the plaintext space to be a public ring in which it is possible to sample elements uniformly at random. It can be applied on BGN-F [3, page 24] after changing the plaintext space from the set $[\![0, n-1]\!]$ to the ring $\mathbb{Z}_n$ where n is a small integer[2]. This gives birth to our scheme BGN-F-CF, an efficient and semantically-secure homomorphic public-key pairing-based encryption scheme that allows the evaluation of a multivariate degree-4 polynomial on ciphertexts over $\mathbb{Z}_n$.

2.2 Notation

Our scheme BGN-F-CF is described thoroughly in Section 3. To help the reader understand its mathematical construction, we provide in this section a brief and high-level description of the underlying BGN [1] scheme and the impact of the modifications due to the application of Freeman [2] and Catalano-Fiore [3] improvements. As notation used in these different papers is not the same, we need to rewrite some parts with uniformed notation.

Before going further, we need to define the notion of ciphertext level, which is very important in these schemes. Figure 1 indicates which operations are permitted on ciphertexts in BGN-F-CF. This correlates with the ciphertext level. A ciphertext, on which no homomorphic operation has been made, has level 1. A homomorphic multiplication produce a ciphertext of level > 1.

* G, H, G_T are abelian groups.
* π_1, π_2, π_T are endomorphisms of G,H,G_T.
* $G_1 \subset \mathrm{Ker}(\pi_1), H_2 \subset \mathrm{Ker}(\pi_2)$ are cyclic groups.
* q_1, q_2, r are prime numbers.

[1] k is the embedding degree with respect to a prime number r such that $r \mid \#E(\mathbb{F}_p)$ and $r \nmid p$ (*i.e.* the smallest integer such that $p^k = 1 \mod r$). In BGN-F and BGN-F-CF, r is the group order of $\mathbb{G}$ and $\mathbb{H}$. In our implementation, $r = \#E(\mathbb{F}_p)$ since Barreto-Naerhig curves have prime order.

[2] In all these variants of BGN, decryption asks to compute discrete logarithms in groups of large prime order r. Thus, plaintext space size n should vary logarithmically with r to have a polynomial-time in n decryption algorithm such as Pollard's kangaroo algorithm [1, page 4].

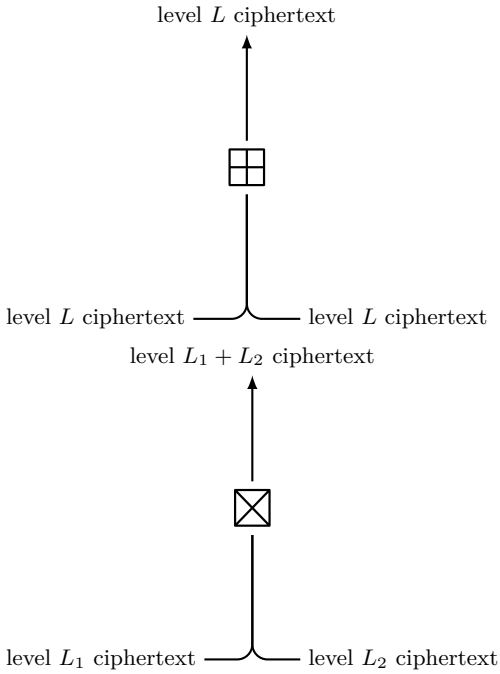

Fig. 1 Homomorphic operations in BGN-F-CF on ciphertexts in the same space. $L \leq 4$, $L_1 \leq 2$, $L_2 \leq 2$.

* e is a pairing.
* E is an elliptic curve.
* $[\![a, b]\!] \coloneqq [a, b] \cap \mathbb{N}$.
* $\mathcal{P}$ is the plaintext space.
* n is a strictly positive integer.
* $\mathbb{Z}_n$ is the ring of integers modulo n.
* $\mathbb{F}_2$ is the two-element finite field.
* The operator $\xleftarrow{\$}$ refers to a random draw according to a uniform distribution.
* $<a>$ the subgroup generated by the element a.
* (a, b) is the concatenation of elements a and b.
* L is the ciphertext level.
* A is the maximal number of level L additions between any input and any output of a circuit to obtain a level L ciphertext. We call it for short, the *L-additive depth*.
* Enc and Dec are encryption and decryption algorithms[3].
* $\text{Enc}^{(2)}(m)$ is a level 2 ciphertext of a message m obtained with BGN-F scheme.
* $\text{End}(G)$ is a endomorphism of a group G.
* Homomorphic addition and multiplication are respectively denoted $\boxplus$ and $\boxtimes$.

In the following, we give generic description of keys and ciphertext space in the three variants: BGN, BGN-F, BGN-F-CF. This helps to distinguish them quickly.

[3] In the family of BGN variants, multiplication and even certain addition modify ciphertext space. For this reason, decryption algorithm and homomorphic operations differ for each level.

2.2.1 BGN

- Public-key **pk**: $(q_1 q_2, G, G_T, e, u, v)$
  - G and G_T are cyclic groups of order $q_1 q_2$.
  - u is a generator of G.
  - v is a generator of subgroup of G of order q_1.
  - e is a symmetric pairing:

$$e : G \times G \mapsto G_T.$$

- Secret-key **sk**: q_1 is a prime number.
- Plaintext space $\mathcal{P}$ is the set $[\![0, n]\!]$: $n < q_2$.

Assumption	Subgroup decision problem on G	
Public-key **pk**	$(q_1 q_2, G, G_T, e, u, v)$ G and G_T are cyclic groups of order $q_1 q_2$	
Secret-key **sk**	q_1	
Plaintext space $\mathcal{P}$	$[\![0, n]\!]$: $n < q_2$	
Ciphertext spaces	level 1	G
	level 2	G_T

Table 1 Concise description of BGN scheme.

2.2.2 BGN-F

- Public-key **pk**: $(G, G_1, H, H_2, G_T, e, u, v)$
  - $G = \mathbb{G}^2$, $H = \mathbb{H}^2$, $G_T = \mathbb{G}_T^4$.
  - $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of prime order r .
  - $u \xleftarrow{\$} G$.
  - $v \xleftarrow{\$} H$.
  - e is an asymmetric pairing:

$$e : G \times H \mapsto G_T.$$

  - e is projecting, it means:

$$\pi_T(e(g, h)) = e\big(\pi_1(g), \pi_2(h)\big),\ \forall (g, h) \in G \times H.$$

- Secret-key **sk**: (π_1, π_2, π_T).
- Plaintext space $\mathcal{P}$ is the set $[\![0, n]\!]$: $n < \log r$.

The abstract framework proposed by Freeman[2, page 46] is generic. It converts a cryptosystem over composite order groups, such as BGN scheme, into a scheme over prime order groups. In our definition of BGN-F, we choose to employ DDH assumption on prime order groups as in [2, page 57]. In this particular setting, the subgroups G_1 and H_2 are cyclic (rank $l = 1$) and the pairing is necessarily asymmetric else security is not

Assumption	DDH on $\mathbb{G}$ and $\mathbb{H}$	
Public-key **pk**	$(G, G_1, H, H_2, G_T, e, u, v)$ $G = \mathbb{G}^2$, $H = \mathbb{H}^2$, $G_T = \mathbb{G}_T^4$ $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order r	
Secret-key **sk**	(π_1, π_2, π_T)	
Plaintext space $\mathcal{P}$	$[\![0, n]\!]$: $n < \log r$	
Ciphertext spaces	level 1	$G \times H$
	level 2	G_T

Table 2 Concise description of BGN-F scheme.

ensured. Such asymmetric pairing is known over ordinary curves whereas BGN uses a symmetric pairing which only exists over supersingular curves. If we used the l-linear assumption[2, page 49] with $l > 1$ on the prime groups, we could employ a symmetric pairing. But, in this case, at standard security levels, the groups $\mathbb{G}$ and $\mathbb{H}$ would be too large since embedding degree k would be too small. We would have $k \leq 6$ [35, page 3]. This would imply that cryptographic operation require much more computations, time and memory in our final scheme, BGN-F-CF. Embedding degree is discussed in Section 5 for more information.

2.2.3 BGN-F-CF

In BGN-F-CF scheme, the couple of keys $(\mathbf{pk}, \mathbf{sk})$ has the same description than in BGN-F scheme (see Section 2.2.2). On the other hand, the plaintext space has a ring structure and the ciphertext space is more complex. Notice, the latter depends on the former for level 1 and level 2 ciphertexts, as indicated in Table 3.

- Plaintext space $\mathcal{P}$ is the ring $\mathbb{Z}_n$: $n < \log r$.

3 Generic description of BGN-F-CF

Catalano-Fiore generalized construction [3, page 26] permits to increment the multiplicative depth of arithmetic circuits that a homomorphic scheme can evaluate over ciphertexts. Circuit multiplicative depth is the maximum number of multiplicative gates between an input and an output in the circuit. In other words, the generalized construction permits to evaluate the product of any ciphertexts obtained with the original scheme. In BGN-F-CF scheme, it permits to evaluate a multivariate polynomial of degree ≤ 4 over ciphertexts. Degree is limited to 2 in the original scheme BGN-F. The generalized construction applies on a *public-space* homomorphic scheme. It is a homomorphic scheme which satisfies the following conditions on plaintext space $\mathcal{P}$:

Assumption	DDH on $\mathbb{G}$ and $\mathbb{H}$	
Public-key **pk**	$(G, G_1, H, H_2, G_T, e, u, v)$ $G = \mathbb{G}^2$, $H = \mathbb{H}^2$, $G_T = \mathbb{G}_T^4$ $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ are groups of order r	
Secret-key **sk**	(π_1, π_2, π_T)	
Plaintext space $\mathcal{P}$	$\mathbb{Z}_n$: $n < \log r$	
Ciphertext spaces	level 1	$\mathcal{P} \times G \times H$
	level 2	$\mathcal{P} \times G_T$
	level 3	$G_T \times (G \times G_T)^{1+A}$ $G_T \times (H \times G_T)^{1+A}$ $G_T \times (G_T \times G)^{1+A}$ $G_T \times (G_T \times H)^{1+A}$
	level 4	$G_T \times (G_T \times G_T)^{1+A}$

Table 3 Concise description of BGN-F-CF scheme.

- $\mathcal{P}$ is public.
- $\mathcal{P}$ is a finite commutative unitary ring.
- $\mathcal{P}$ is efficiently samplable uniformly at random.

It can require to adapt a homomorphic scheme. It is the case for BGN and BGN-F since in these cryptosystems, $\mathcal{P}$ is a set without algebraic structure.
We apply Catalano-Fiore generalized transformation on a public-space version [3, page 2] of BGN-F scheme. The scheme BGN-F-CF is defined relatively to the encryption algorithm Enc and the homomorphic addition in BGN-F. Let us recall their definition.
To begin, in the generic case, we choose to consider multiplicative groups. The encryption is probabilistic, the client draws at random $u_1 \xleftarrow{\$} G_1$ and $v_1 \xleftarrow{\$} H_2$. An encryption of a plaintext m is:

$$\mathrm{Enc}(m) = (u^m u_1, v^m v_1) \in G \times H.$$

For the addition, the server draws at random $u_1 \xleftarrow{\$} G_1$ and $v_1 \xleftarrow{\$} H_2$. c_1 and c_2 being two ciphertexts, the homomorphic addition is defined as below,

$$c_1 \boxplus c_2 = \begin{cases} c_1 c_2 u_1 & \text{if } c_1, c_2 \in G. \\ c_1 c_2 v_1 & \text{if } c_1, c_2 \in H. \\ c_1 c_2 e\big(u, v_1\big) e\big(u_1, v\big) & \text{if } c_1, c_2 \in G_T. \end{cases}$$

We do not describe multiplication of a ciphertext by a plaintext. The operation is described in [3, page 10] for a linearly-homomorphic cryptosystem compatible with Catalano-Fiore transformation. Indeed, our instantiation manages binary messages and in this case, such operation can be done otherwise, if necessary. We do not describe either re-randomization operation [3, page 27] as we do not implement it. We place ourselves in

a client-server model, where we only require semantic security on the message, to protect client data. Rerandomization is one solution to obtain an additional property: circuit privacy. It is useful in scenarios where the server, which operates homomorphic computations, requires to hide any information on the circuit, beyond ciphertexts.
Now, we give the description of our scheme in the formal steps of key generation, encryption, decryption, and how homomorphic operations are performed.

3.1 Key generation

We generate abelian groups G, H, G_T equipped with an asymmetric and projecting pairing $e : G \times H \mapsto G_T$. We consider group endomomorphisms π_1, π_2, π_T over G, H, G_T respectively. Then we generate subgroups G_1 (resp. H_2) in the kernel of π_1 (resp. π_2). The endomorphisms serve as secret-key, while every groups and the pairing serve as public-key. Key generation is run as:

1. Generate a tuple $(G, G_1, H, H_2, G_T, \pi_1, \pi_2, \pi_T, e)$ such that $e\big(\pi_1(g), \pi_2(h)\big) = \pi_T(e\big(g, h\big)\big)$, $\forall g \in G, h \in H$.
2. Choose $u \xleftarrow{\$} G$ and $v \xleftarrow{\$} H$.
3. **sk** is (π_1, π_2, π_T).
4. **pk** is $(G, G_1, H, H_2, G_T, e, u, v)$.

3.2 Encryption

Consider a client with a message $m \in \mathcal{P}$. Catalano-Fiore transformation introduces a random element $b \in \mathcal{P}$ during encryption process. Encryption is probabilistic and makes use of random subgroup elements. The client computes $a = m - b$ and uses BGN-F to encrypt b. The ciphertext c is the concatenation of a and of the encryption of $m - b$.

1. Choose a message m in $\mathcal{P}$ with $n \leq \log(r)$.
2. Choose $b \xleftarrow{\$} \mathcal{P}$.
3. Compute $a = m - b$.
4. Choose $u_1 \xleftarrow{\$} G_1$, $v_1 \xleftarrow{\$} H_2$.
5. The level 1 ciphertext is:

$$c = (a, u^b u_1, v^b v_1) \in \mathcal{P} \times G \times H.$$

During encryption, in BGN-F, the computations are duplicated in the two groups G and H. In addition, depending on the circuit that the server evaluates over ciphertexts, it can omit or not the computation in G or H. Subsequently, for easy reading, we consider only two components ciphertexts in $\mathcal{P} \times G$ or $\mathcal{P} \times H$.

3.3 Homomorphic operation

Consider a server with two ciphertexts c_1 and c_2. Take $c_1 = (a_1, \beta_1)$ and $c_2 = (a_2, \beta_2)$. The server computes a ciphertext $c = (\alpha, \beta)$. Operations are level-dependant as indicated in Figure 1.

3.3.1 Homomorphic Addition.

• Addition of level 1 or level 2 ciphertexts simply consists in adding the two components. This leads to:

$$(a, \beta) = (a_1 + a_2, \beta_1 \boxplus \beta_2),$$

where the symbol $\boxplus$ refers to the homomorphic addition in BGN-F.
• Addition of level 3 or level 4 ciphertexts is done in two simple steps:

1. Add the first components $\alpha = \alpha_1 \boxplus \alpha_2$.
2. Concatenate the second components $\beta = (\beta_1, \beta_2)$.

Concatenation saves computation cost but increases ciphertext size.

3.3.2 Homomorphic Multiplication

• Multiplication of two level 1 ciphertexts takes as input ciphertexts in distinct spaces since it requires to evaluate an asymmetric pairing. The server computes the product ciphertext in $\mathcal{P} \times G_T$ as follows:

1. Take one ciphertext:

$$c_1 = (a_1, \beta_1) \in \mathcal{P} \times G.$$

2. Take another ciphertext:

$$c_2 = (a_2, \beta_2) \in \mathcal{P} \times H.$$

3. Choose $b_1, b_2, s \xleftarrow{\$} \mathcal{P}$.
4. Choose again $u_1 \xleftarrow{\$} G_1$, $v_1 \xleftarrow{\$} H_2$.
5. The level 2 ciphertext is $(a, \beta) \in \mathcal{P} \times G_T$ with:

$$a = a_1 a_2 - s.$$

$$\beta = e\big(\beta_1, \beta_2\big) e\big(u, v_1\big) e\big(u_1, v\big) \boxplus \beta_2^{b_1} \boxplus \beta_1^{b_2} \boxplus \mathrm{Enc}^{(2)}(s).$$

Above, the term $\mathrm{Enc}^{(2)}(s)$ is a level 2 ciphertext. Encryption of s gives a level 1 ciphertext. To increment the level of a ciphertext, the server multiplies it homomorphically by an encryption of 1.
• Multiplication of one level 1 ciphertext with one level 2 ciphertext outputs a level 3 ciphertext as shown in Figure 1. Suppose c_1 is level 1 and c_2 is level 2. For level 3 ciphertexts, four ciphertext spaces are indicated in Table 1. Indeed, it depends on:

- c_1 belongs to $\mathcal{P} \times G$ or $\mathcal{P} \times H$.
- c_1 is the left operand or the right one.

• Multiplication of two level 2 ciphertexts in $\mathcal{P} \times G_T$ returns a level 4 ciphertext in G_T^3.

For the two last cases, the first component α belongs to the target group G_T and the server computes the product ciphertext (α, β) as follows:

$$\alpha = \mathrm{Enc}(a_1 a_2) \boxplus \beta_2^{a_1} \boxplus \beta_1^{a_2},$$

$$\beta = (\beta_1, \beta_2).$$

3.4 Decryption

Consider a client with a ciphertext $c = (a, \beta)$. It searches to recover the message $m \in \mathcal{P}$.
• Decryption of a level 1 or a level 2 ciphertext.
In this case, it applies the private key which consists in endomorphisms π_1, π_2 and π_T on the element β. This element β contains a blinding element which is in subgroups G_1, G_2, G_T. The blinding element vanishes when the client applies the endomorphism since the subgroups are contained in the kernel of endomorphisms. The client computes a discrete logarithm. This element β is a ciphertext in BGN-F. This procedure is the decoding algorithm Dec in BGN-F. The extra part in BGN-F-CF asks the client to add a, the first component of the ciphertext to get the message m. In other words $m = a + \mathrm{Dec}(\beta)$. More precisely, this gives:

$$m = \begin{cases} a + \log_{\pi_1(u)} \pi_1(\beta) & \text{if } c \in \mathcal{P} \times G. \\ a + \log_{\pi_2(v)} \pi_2(\beta) & \text{if } c \in \mathcal{P} \times H. \\ a + \log_{\pi_T(e(u,v))} \pi_T(\beta) & \text{if } c \in \mathcal{P} \times G_T. \end{cases}$$

• Decryption of a level 3 or a level 4 ciphertext
We treat the general case where the client receives a ciphertext $c = (\alpha, \beta)$ obtained with A additions of different level L ciphertexts with $3 \leq L \leq 4$. Actually, α is a level 2 ciphertext in BGN-F and β is composed of $2 \times (1 + A)$ ciphertexts in BGN-F.

$$\beta \coloneqq (\beta_{1,1}, \beta_{2,1}, \beta_{1,2}, \beta_{2,2}, \ldots, \beta_{1,1+A}, \beta_{2,1+A}).$$

where $\forall\ i, j \in [\![1, 1 + A]\!]$, $\beta_{i,j}$ is either a level 1 ciphertext, if $L = 3$, or a level 2 ciphertext, if $L = 4$.
The decryption of c calls out $2 \times A + 3$ times, the decryption algorithm of BGN-F scheme, denoted Dec and described in the previous item. To obtain the message $m \in \mathcal{P}$, the process works as follows:

$$m = \mathrm{Dec}(\alpha) + \sum_{i=1}^{1+A} \mathrm{Dec}(\beta_{1,i})\,\mathrm{Dec}(\beta_{2,i}).$$

4 Instantiation of BGN-F-CF

In this section, we describe our BGN-F-CF implementation. Security and library choices are discussed in Section 5.
First of all, in our implementation we decided to restrain to the evaluation of polynomials with binary coefficients, in order to have an efficient decryption algorithm. Moreover, independently of BGN-F-CF computations, many treatments are applied on binary data. For instance, to compare integers or to output different results depending on an integer value, in the encrypted domain, encoding is performed before encryption and evaluation (operations on encrypted data). In our implementation of BGN-F-CF, encoding simply implies to decompose integers into bits, whereas it is a "highly non-trivial task" in SEAL library [36, page 15] and other ring-learning-with-errors based scheme implementations, where it is necessary to encode information into a polynomial before encrypting [16, page 13]. After that, encryption is performed bitwise.

4.1 Initial setting

The groups G and H are defined from an elliptic curve E with equation $y^2 = x^3 + 3$ over a finite field. The curve is in the Barreto-Naerhig family of pairing-friendly curves. We remind that pairing-friendly curves [35, page 11] are defined by a triplet $(p(x), r(x), t(x))$ of univariate polynomial and a parameter x_0. Evaluations of these polynomials at x_0 enable us to define curve parameters: p, r and t. Where, p is size of the field on which E is defined, r is the number of $\mathbb{F}_p$-rational points on E and t is the trace of E over $\mathbb{F}_p$. We chose to use the DCLXVI library [37] in our implementation to compute efficiently parings in BGN-F-CF. This library defines a subfamily of Barreto-Naerhig curves where the parameter x_0 is a cube integer.
The library uses $x_0 = y_0^3$ and $y_0 = 1868033$. These parameters permit to define the triplet (p, r, t), as follows:

$$p = p(x_0), r = r(x_0), t = t(x_0).$$

$$\begin{cases} p(x) = 36x^4 + 36x^3 + 24x^2 + 6x + 1. \\ r(x) = 36x^4 + 36x^3 + 18x^2 + 6x + 1. \\ t(x) = 6x^2 + 1. \end{cases}$$

The parameters p and r are 256-bits prime integers, the trace t of E over $\mathbb{F}_p$ is a 128-bits integer. 12 is the embedding degree of r *i.e.* the one of the group $E(\mathbb{F}_p)[r] = E(\mathbb{F}_p)$ since $r = \#E(\mathbb{F}_p)$ is prime. This

implies that the group of r^{th} roots of unity μ_r is included in $\mathbb{F}_{p^{12}}^{\times}$ and that the r-torsion subgroup $E[r]$ is defined over $\mathbb{F}_{p^{12}}$. The embedding degree is particularly important when we use a pairing over elliptic curves, it is discussed in Section 5.
Let us describe briefly the pairing. We need first to define the Frobenius endomorphism π on E as:

$$\begin{aligned} \pi : \quad E \quad &\to \quad E \\ (x, y) &\mapsto (x^p, y^p). \end{aligned}$$

In our implementation, we compute the optimal Ate pairing e_{OA}. The algorithm we use to compute it is given in [37, page 4]. It is such that:

$$e_{OA} \colon E(\mathbb{F}_p) \times (E[r] \cap \mathrm{Ker}(\pi - p)) \to \mu_r.$$

In BGN-F-CF, we instantiate the groups as:

$$\begin{cases} \mathbb{G} &= E(\mathbb{F}_p). \\ \mathbb{H} &= E[r] \cap \mathrm{Ker}(\pi - p) \subseteq E(\mathbb{F}_{p^{12}})[r]. \\ \mathbb{G}_T &= \mu_r. \end{cases}$$

They are publicly known and offer both efficient group operation and random sampling according to a uniform distribution. We can also specify the pairing e over direct groups $G = E(\mathbb{F}_p)^2$ and $H = \big(E[r] \cap \mathrm{Ker}(\pi - p)\big)^2$ which maps to $G_T = \mu_r^4$, as follows:

$$\begin{aligned} e\big((g_1, g_2)(h_1, h_2)\big) \mapsto \big(&e_{OA}(g_1, h_1), e_{OA}(g_1, h_2), \\ &e_{OA}(g_2, h_1), e_{OA}(g_2, h_2)\big). \end{aligned}$$

We choose randomly generators g and h of $E(\mathbb{F}_p)$ and $E[r] \cap \mathrm{Ker}(\pi - p)$.

$$\begin{aligned} g &\xleftarrow{\$} E(\mathbb{F}_p) &&: \mathrm{ord}(g) = r. \\ h &\xleftarrow{\$} E[r] \cap \mathrm{Ker}(\pi - p) &&: \mathrm{ord}(h) = r. \end{aligned}$$

We also compute randomly generators u and v of the direct groups $E(\mathbb{F}_p)^2$ and $(E[r] \cap \mathrm{Ker}(\pi - p))^2$.

$$\begin{aligned} u &\xleftarrow{\$} E(\mathbb{F}_p)^2. \\ v &\xleftarrow{\$} (E[r] \cap \mathrm{Ker}(\pi - p))^2. \end{aligned}$$

4.2 Key generation

We generate the public-private key pair $(\mathbf{pk}, \mathbf{sk})$ as,

$$\mathbf{pk} = \big((i_1 g, j_1 g), (i_2 h, j_2 h), \mu_r, e, u, v\big).$$

$$\mathbf{sk} = (\pi_1, \pi_2, \pi_T).$$

At first, we perform random draws:

$$\begin{aligned} &i_1, j_1, k_1, l_1, i_2, j_2, k_2, l_2 \xleftarrow{\$} \mathbb{F}_p \\ &\qquad \text{until we have: } i_1 l_1 - j_1 k_1 = i_2 l_2 - j_2 k_2 = 1. \end{aligned}$$

The private key $\mathbf{sk}$ is described by endomorphisms π_1, π_2 and π_T. We define first, π_1 and π_2 such that:
$\pi_1 \in \mathrm{End}(E(\mathbb{F}_p)^2), \pi_2 \in \mathrm{End}(E[r] \cap \mathrm{Ker}(\pi - p)^2)$.

$$\begin{aligned} \pi_1(x, y) &= (-j_1 k_1 x + i_1 k_1 y, -j_1 l_1 x + i_1 l_1 y). \\ \pi_2(x, y) &= (-j_2 k_2 x + i_2 k_2 y, -j_2 l_2 x + i_2 l_2 y). \end{aligned}$$

To specify the endomorphism π_T we choose to reuse a notation given by Freeman[2, page 52], more compact than the usual one. Let $\mathcal{M} = (m_{i,j})$ be an n-order matrix over $\mathbb{F}_p$. Its coefficients are taken columnwise to define:

$$\gamma^{\mathcal{M}} \coloneqq (\prod_{i=1}^{n} \gamma_i^{m_{i1}}, \ldots, \prod_{i=1}^{n} \gamma_i^{m_{in}})$$

with $\gamma = (\gamma_1, \ldots, \gamma_n)$ in a direct group of n groups.
Let us consider the tensor product $\mathcal{A} \otimes \mathcal{B}$ with:

$$\mathcal{A} = \begin{pmatrix} -j_1 k_1 & -j_1 l_1 \\ i_1 k_1 & i_1 l_1 \end{pmatrix}, \quad \mathcal{B} = \begin{pmatrix} -j_2 k_2 & -j_2 l_2 \\ i_2 k_2 & i_2 l_2 \end{pmatrix}.$$

It is a matrix[4] of order 4 over $\mathbb{F}_p$. Finally, we get:

$$\begin{aligned} \pi_T : \mu_r^4 &\to \quad \mu_r^4 \\ x \quad &\mapsto x^{\mathcal{A} \otimes \mathcal{B}}. \end{aligned}$$

As a toy example, we can fold π_1 under this compact form.

$$\begin{aligned} \pi_1 : E(\mathbb{F}_p)^2 &\to E(\mathbb{F}_p)^2 \\ x \quad &\mapsto \quad x^{\mathcal{A}}. \end{aligned}$$

The public key $\mathbf{pk}$ is described by generators u and v of direct groups G and H, generators of their subgroups $G_1 = <(i_1 g, j_1 g)>$ and $H_2 = <(i_2 h, j_2 h)>$, the pairing e and the multiplicative group $\mu_r \subset \mathbb{F}_{p^{12}}$.

4.3 Encryption of a single bit

The binary plaintext m is transformed into the ciphertext c as follows:

$$b \xleftarrow{\$} \mathbb{F}_2.$$

$$a = m - b.$$

[4] We can divide it into 4 matrices of order 2. The $(i,j)^{\text{th}}$ block is equal to $a_{i,j}\mathcal{B}$ with $\mathcal{A} = (a_{i,j})_{i,j \in \{1,2\}}$.

We compute random elements u_1 and v_1 in the subgroups $<(i_1g, j_1g)>$ and $<(i_2h, j_2h)>$.

$$u_1 \xleftarrow{\$} <(i_1g, j_1g)> \leq E(\mathbb{F}_p)^2$$
$$v_1 \xleftarrow{\$} <(i_2h, j_2h)> \leq (E[r] \cap \mathrm{Ker}(\pi - p))^2$$

$$c = (a, bu + u_1, bv + v_1) \in \mathbb{F}_2 \times E(\mathbb{F}_p)^2 \times (E[r] \cap \mathrm{Ker}(\pi - p))^2.$$

4.3.1 Curve mode and twist mode

To define homomorphic operations and decryption, we reduce the ciphertext to two components instead of three. We only need the first and one of the two last components. We speak of *curve mode* and *twist mode*, this helps readability.
Curve mode is when:

$$c = (a, bu + u_1) \in \mathbb{F}_2 \times E(\mathbb{F}_p)^2.$$

Twist mode is when:

$$c = (a, bv + v_1) \in \mathbb{F}_2 \times (E[r] \cap \mathrm{Ker}(\pi - p))^2.$$

Let us give an example. To obtain the product of two level 1 ciphertexts, we need one ciphertext in curve mode and one other in twist mode. Indeed, we employ an asymmetric pairing defined over a group product of points on a curve and points on the twist. More precise information on this topic is given in Subsection 4.4.2.

4.4 Homomorphic operations

4.4.1 Addition of ciphertexts

Homomorphic additions are defined with respect to ciphertext levels.
• Addition of level L ciphertexts with $1 \leq L \leq 2$ Let, $c_1 = (a_1, \beta_1)$ and $c_2 = (a_2, \beta_2)$ be two ciphertexts as input, with the same level $1 \leq L \leq 2$ and $a_1, a_2 \in \mathbb{F}_2$. On output, there is one level L ciphertext $c = (a, \beta)$. The three ciphertexts are in the same space.
For the second component of the ciphertexts, three configurations are possible,

- $\beta_1, \beta_2 \in E(\mathbb{F}_p)^2$.
- $\beta_1, \beta_2 \in (E[r] \cap \mathrm{Ker}(\pi - p))^2$.
- $\beta_1, \beta_2 \in \mu_r^4$.

The first component of the resultant ciphertext is computed as,

$$a = a_1 + a_2 \ .$$

Then to compute the second component of the resultant ciphertext, we redo a random uniform draw for u_1 and v_1.

$$u_1 \xleftarrow{\$} <(i_1g, j_1g)>, \ v_1 \xleftarrow{\$} <(i_2h, j_2h)>.$$

Then depending on the configuration mentioned above, we compute β respectively.

- $\beta = \beta_1 + \beta_2 + u_1$ if $\beta_1, \beta_2 \in E(\mathbb{F}_p)^2$.
- $\beta = \beta_1 + \beta_2 + v_1$ if $\beta_1, \beta_2 \in (E[r] \cap \mathrm{Ker}(\pi - p))^2$.
- $\beta = \beta_1\beta_2 e\big(u, v_1\big)e\big(u_1, v\big)$ if $\beta_1, \beta_2 \in \mu_r^4$.

• Addition of level L ciphertexts with $3 \leq L \leq 4$ Let, there are two level L ciphertexts $c_1 = (\alpha_1, \beta_1)$ and $c_2 = (\alpha_2, \beta_2)$, with $L \in [\![3, 4]\!]$. The two ciphertexts are in the same ambient space. On output, there is one level L ciphertext $c = (\alpha, \beta)$.
To simplify and benefit group commutativity, if one factor is a level 1 ciphertext, it is the left operator. In addition, to save memory, the level 1 ciphertext is computed in curve mode. Indeed, a level 1 ciphertext in twist mode is two times bigger than in curve mode, as indicated in Table 6. These restrictions do not limit the family of polynomials that we can evaluate over ciphertexts. No multiplication is defined with a level 3 ciphertext. We can only operate addition between level 3 ciphertexts sharing the same structure. In our implementation, it is:

$$\mu_r^4 \times (E(\mathbb{F}_p)^2 \times \mu_r^4)^{1+A}.$$

We compute the first component of the resultant ciphertext as,

$$\alpha = \alpha_1 \boxplus \alpha_2 = \alpha_1\alpha_2 e\big(u, v_1\big)e\big(u_1, v\big).$$

For every instance, $\alpha, \alpha_1, \alpha_2 \in \mu_r^4$.
The second component of the resultant ciphertext is the concatenation of the second components of the inputs.

$$\beta = (\beta_1, \beta_2).$$

Recall $L = 3$ or $L = 4$. Each addition and multiplication (see Section 4.4.2) to obtain a level L ciphertext, expands the ciphertext size. For this reason, we operate a limited number of such additions in practice.
For a ciphertext of level $3 \leq L \leq 4$, obtained after A additions of level $3 \leq L \leq 4$, there are two cases:

- $\beta \in (E(\mathbb{F}_p)^2 \times \mu_r^4)^{1+A}$.
- $\beta \in (\mu_r^4 \times \mu_r^4)^{1+A}$.

4.4.2 Multiplication of ciphertexts

As in addition, the homomorphic multiplication is level dependent.

• Multiplication of level 1 ciphertexts

The pairing is used to multiply level 1 ciphertexts. It is asymmetric and thus requires a ciphertext in curve mode and another in twist mode. Let, c_1 and c_2 are two input ciphertexts of level 1.

$$c_1 = (a_1, \beta_1) \in \mathbb{F}_2 \times E(\mathbb{F}_p)^2.$$
$$c_2 = (a_2, \beta_2) \in \mathbb{F}_2 \times \big(E[r] \cap \mathrm{Ker}(\pi - p)\big)^2.$$

We draw $b_1, b_2, s \xleftarrow{\$} \mathbb{F}_2$.

For $i = 1$ and $i = 2$, c_i is an encryption of $m_i \in \mathbb{F}_2$ and $a_i = m_i - b_i$.

The first component of the resultant ciphertext is computed as,

$$a = a_1 a_2 - s.$$

To compute the second component of the resultant ciphertext, at first, we redo a random uniform draw for u_1 and v_1.

$$u_1 \xleftarrow{\$} <(i_1 g, j_1 g)>,\ v_1 \xleftarrow{\$} <(i_2 h, j_2 h)>.$$

We split up the computation of β in order to explain how the formula is obtained. The reader can skip this paragraph and only retain the final formula for a practical usage. We recall the operator $\boxplus$ refers to a homomorphic addition with the scheme BGN-F.

Then, we proceed to compute β as,

$$\beta = e\big(\beta_1, \beta_2\big)e\big(u, v_1\big)e\big(u_1, v\big) \boxplus a_1\beta_2 \boxplus a_2\beta_1 \boxplus \mathrm{Enc}^{(2)}(s).$$

Let μ_r be the subgroup of r^{th}-roots of unity in $\mathbb{F}_{p^{12}}$. The first term belongs to μ_r^4. Note that, the level should be the same for all terms[5].

After drawing: $u_2, u_3, u_4 \xleftarrow{\$} <(i_1 g, j_1 g)>$,

and $v_2, v_3, v_4 \xleftarrow{\$} <(i_2 h, j_2 h)>$, we compute β as

$$\begin{aligned}\beta = {} & e\big(\beta_1, \beta_2\big)e\big(u, v_1\big)e\big(u_1, v\big) \\ & \boxplus e\big(\mathrm{Enc}(1), a_1\beta_2\big)e\big(u, v_2\big)e\big(u_2, v\big) \\ & \boxplus e\big(a_2\beta_1, \mathrm{Enc}(1)\big)e\big(u, v_3\big)e\big(u_3, v\big) \\ & \boxplus e\big(\mathrm{Enc}(1), \mathrm{Enc}(s)\big)e\big(u, v_4\big)e\big(u_4, v\big).\end{aligned}$$

Using bilinearity, we can simplify this expression. In practice, it is not useful to define $u_2, u_3, u_4, v_2, v_3, v_4$. On the other hand, it is useful to understand how we obtain the following formula:

$$\begin{aligned}\beta = {} & e\big(\beta_1, \beta_2\big)e\big(\mathrm{Enc}(1), a_1\beta_2 + \mathrm{Enc}(s)\big) \\ & e\big(a_2\beta_1, \mathrm{Enc}(1)\big)e\big(u, v_1\big)e\big(u_1, v\big).\end{aligned}$$

We compute 5×4 (optimal Ate) pairings to get a level 2 ciphertext.

• Multiplication of ciphertexts to obtain a level L ciphertext with $3 \leq L \leq 4$

On input, there are two ciphertexts (a_1, β_1) and (a_2, β_2), with levels $L_1, L_2 \in [\![1, 2]\!]$ verifying the condition[6]:

$$3 \leq L_1 + L_2 \leq 4$$

and $a_1, a_2 \in \mathbb{F}_2$. On output, there is one level L ciphertext (α, β) with $L = L_1 + L_2$. The symbol $\boxplus$ refers to a homomorphic addition with the scheme BGN-F.

Let us recall that we can only add ciphertexts of same level, as explained in Section 4.4.1. To compute α, we should get level 2 terms.

A level 3 ciphertext is such that:

$$\begin{cases} \alpha = e\big(\mathrm{Enc}(a_1 a_2), \mathrm{Enc}(1)\big)\beta_2^{a_1} \\ e\big(a_2\beta_1, \mathrm{Enc}(1)\big)e\big(u, v_1\big)e\big(u_1, v\big). \\ (\beta_1, \beta_2) \in E(\mathbb{F}_p)^2 \times \mu_r^4. \end{cases}$$

A level 4 ciphertext is such that:

$$\begin{cases} \alpha = e\big(\mathrm{Enc}(a_1 a_2), \mathrm{Enc}(1)\big)\beta_2^{a_1}\beta_1^{a_2}e\big(u, v_1\big)e\big(u_1, v\big). \\ (\beta_1, \beta_2) \in \mu_r^4 \times \mu_r^4. \end{cases}$$

The first two cases permit to evaluate the same products since the multiplication is commutative over $\mathbb{F}_2$. We choose to limit ourselves to the first case where the first ciphertext has level 1, and the second ciphertext has level 2. Once again, in the first case, we restrict, for convenience, β in the direct group $(E(\mathbb{F}_p)^2 \times \mu_r^4)$. In every instance, $\alpha \in \mu_r^4$. The number of successive multiplications is limited to one because the scheme BGN-F-CF evaluates ciphertexts up to level 4. Notice, the computation of a level 3 ciphertext needs the computation of 4×4 pairings instead of 3×4 pairings for the computation of a level 4 ciphertext. The additional pairings are an extra part of the computation cost when we multiply two ciphertexts of different levels.

[5] If it is not the case, we multiply homomorphically the other terms by Enc(1), an encryption of bit 1. More generally, this is applied several times when we compute the sum of ciphertexts with several levels of difference.

[6] We can obtain level 4 ciphertexts with product of two level 2 ciphertexts but no product between a level 1 ciphertext and a level 3 ciphertext is defined.

4.5 Decryption

• Decryption of level 1 and 2 ciphertext

The binary case can be rewritten without using discrete logarithm[7]contrarily to the generic case described in Section 3.4. To decrypt a ciphertext $c = (a, \beta)$, we employ the private key which consists in endomorphism π_1, π_2 and π_T:

$$m = \mathrm{Dec}(c) = \begin{cases} a + \dfrac{\pi_1(bu + u_1)}{\pi_1(u)} \text{if } c \in \mathbb{F}_2 \times G. \\ a + \dfrac{\pi_2(bv + v_1)}{\pi_2(v)} \text{if } c \in \mathbb{F}_2 \times H. \\ a + \dfrac{\pi_T(\beta)}{\pi_T(e(u,v))} \text{if } c \in \mathbb{F}_2 \times G_T. \end{cases}$$

Note that the client can precompute the denominator which is independent of the message. This precomputation permits to decrease decryption time in a significant way, for any ciphertext, regardless of its level. As we will see later, decryption of level 3 and level 4 ciphertext uses respectively two and three times, the decryption of level 2 ciphertexts.

• Decryption of level L ciphertext with $3 \leq L \leq 4$

Let us consider a level 3 or level 4 ciphertext $c = (\alpha, \beta)$. Where, α is a level 2 ciphertext.

As explained in Subsection 4.4.2, for level 3 ciphertext, we can restrict to the case:

$$c = (\alpha, \beta) \in \mu_r^4 \times (E(\mathbb{F}_p)^2 \times \mu_r^4)^{1+A}$$

obtained with A additions of different level 3 ciphertexts.

We can now describe β as:

$$\beta \coloneqq (\beta_{1,1}, \beta_{2,1}, \beta_{1,2}, \beta_{2,2}, \ldots, \beta_{1,1+A}, \beta_{2,1+A}).$$

where $(\beta_{1,i})_{i \in [\![1,1+A]\!]}$ is either a family of level 1 ciphertext or level 2 ciphertexts and $(\beta_{2,i})_{i \in [\![1,1+A]\!]}$ is a family of level 2 ciphertext.

We obtain the plaintext $m \in \mathbb{F}_2$, by computing:

$$m = \mathrm{Dec}(\alpha) + \sum_{i=1}^{1+A} \mathrm{Dec}(\beta_{1,i})\,\mathrm{Dec}(\beta_{2,i}).$$

[7]In the equality, the ratio is well-defined if the numerator is a multiple of the denominator. Its value is equal to the scalar factor between the two points, modulo 2. This is the case here since u_1 and v_1 belongs respectively to the kernels of π_1 and π_2 [2, page 58].

4.6 Ciphertext space

Figure 1 indicates which operations are permitted on ciphertexts in BGN-F-CF. This correlates with the ciphertext level. A ciphertext on which no homomorphic operation has been made, has level 1. A homomorphic multiplication produces a ciphertext of level > 1. Table 4 summarizes the different ciphertext spaces according to this notion of ciphertext level as well as circuit multiplicative depth.

Circuit Mult. Depth	Cipher Text Level	Ciphertext space
0	C	$\mathbb{F}_2 \times E(\mathbb{F}_p)^2$
	T	$\mathbb{F}_2 \times (E[r] \cap \mathrm{Ker}(\pi - p))^2$
1	2	$\mathbb{F}_2 \times \mu_r^4$
2	3	$\mu_r^4 \times (E(\mathbb{F}_p)^2 \times \mu_r^4)^{1+A}$
	4	$\mu_r^4 \times (\mu_r^4 \times \mu_r^4)^{1+A}$

Table 4 The different ciphertext spaces according to circuit multiplicative depth. A is the L-additive depth of a circuit which produces a level L ciphertext. The notations C and T refer to a level 1 ciphertext in curve mode and twist mode respectively.

5 Security

BGN-F-CF belongs to the family of pairing-based cryptosystems [7]. We consider pairings over groups, they are bilinear maps that we can compute efficiently and that should offer the targeted level of security. Such pairings are known over elliptic curves and hyperelliptic curves. As performances are not better on the latter [38,39], we chose the first which also benefits from key-size recommendations[8]. Security reduces to the Discrete Logarithm Problem over elliptic curves and over finite fields. The security level should be the same between input groups over elliptic curves and the target groups over finite field. But, the group sizes differ significantly between the two types of groups, since more efficient algorithms are known to solve the Discrete Logarithm over finite fields than over elliptic curves. In BGN-F-CF, we work with asymmetric pairing

$$e : G \times H \mapsto G_T.$$

Groups G, H and G_T are constructed from r-order groups with r prime. G and H are defined using group

[8]see `https://www.keylength.com/`

of points of an elliptic curve defined over $\mathbb{F}_p$. The group G_T is taken as the multiplicative group of the finite field $\mathbb{F}_{p^k}$. In our case, k is the embedding degree of r, since we consider an elliptic curve defined over a prime field[40]. For this reason, the embedding degree k is a curve parameter, which is important for pairing-based cryptography. It impacts the choice of the elliptic curve we use. For a security level of 128 bits, the recommended size for finite fields is 3072 bits and for elliptic curves, the size recommended is 256 bits. In our case, it means that p^k has size 3072 and r has size 256. In addition, p and r should have almost the same size[9] for efficient arithmetic over elliptic curves according to [35,41]. Thus, k should be 12 for security level of 128 bits since $3072 = 12 \times 256$. The curves in the Barreto-Naerhig family reach this optimal value and that is why we chose them. To choose another security level, we can sum up. There are three key parameters: p, r and k at a given security level. Their choice is guided by the motivations given in Table 5. The task is then to find an elliptic curve with such parameters, we do not discuss this here for scope limitation.

Requirements	Reasons
large r	security under ECDH
very large p^k	security under DDH
$r = pk$	same security under ECDH and DDH
$\frac{p}{r} \approx 1$	efficient arithmetic on EC minimize ciphertext size

Table 5 The choice of parameters size in BGN-F-CF, at a fixed security level. Updated recommended values of r and p^k to resist discrete logarithm attacks are available at `https://www.keylength.com`. Recall the Hasse-Weil bound, $|r-p-1| \leq 2\sqrt{p}$.

Let us now discuss the security assumption on which BGN-F-CF rests on. BGN-F-CF scheme is secure under decisional Diffie-Hellman assumption (also called 1-linear assumption) in prime groups $\mathbb{G} = E(\mathbb{F}_p)$, $\mathbb{H} = E[r] \cap \mathrm{Ker}(\pi - p)$ and $\mathbb{G}_T = \mu_r$. This can be reformulated [2, page 46] as secure under generalized subgroup decision in direct groups $G = E(\mathbb{F}_p)^2$, $H = (E[r] \cap \mathrm{Ker}(\pi - p))^2$ and $G_T = \mu_r^4$. Let us recall the definition of this problem.

- 5 distinct groups $G_1 \subset G, H_1 \subset H, G_T$.
- non-degenerate bilinear map $e : G \times H \to G_T$.
- Problem 1: distinguish $x \xleftarrow{\$} G_1$ from $x \xleftarrow{\$} G$.
- Problem 2: distinguish $y \xleftarrow{\$} H_1$ from $y \xleftarrow{\$} H$.

If both problems are computationally infeasible, then generalized subgroup decision assumption holds for (G, G_1, H, H_1, G_T, e).

In BGN-F-CF, the first homomorphic multiplication asks to compute pairings. We chose to compute pairings over elliptic curves in our implementation. Elliptic curve has to be ordinary (*i.e.* $\#E(\mathbb{F}_p) \neq p + 1$) to resist attacks against decisional Diffie-Hellman [2, page 46], contrary to the original BGN scheme where supersingular curves could be used. In this setting, pairing is asymmetric[10], *i. e.* it takes as inputs, elements of two distinct subgroups of an elliptic curve. The pairing image is in an extension field of a prime field of medium characteristic [42, page 2] compared to the extension degree. Moreover, each group involved in the pairing has to be big enough to resist attacks against discrete logarithms since they are the most efficient (known) way to solve decisional Diffie Hellman. More precisely, parameters p, r and t are parametrized by x_0, a cube integer instead of any integer. This has to be taken into account to update parameters using DCLXVI library, after discrete logarithm attacks[11]. The approach of [43] can be used to generate suitable parameters with Barreto-Naerhig curves. The recommended group size given by different academic and private organizations at www.keylength.com according to standard security levels, do not yet take into account most recent security estimations. We would like to point out that our implementation with 256-bits primes p and r ensures today around 100-bits security and not 128-bits security, as it was targeted a few months ago. Indeed, pairing-based cryptography using target field $\mathbb{F}_{p^6}$ and $\mathbb{F}_{p^{12}}$ is affected by Kim-Barbulescu variant of the Number Field Sieve [42]. This forces to re-evaluate parameters if we want to maintain a 128-bits security level. According to [44, page 17], 383-bits primes are required to define a Barreto-Naerhig curve with 128-bits security level whereas [45, page 18] advise for a 461-bit prime. Moreover, the latter indicate, pairing can be computed more efficiently at 128-bit security level with KSS16 curves [46] over a 340-bit base field.

Currently, the best known classical attack against Discrete Logarithm Problem over finite fields is (S)exTNFS algorithm [42], a variant of the number field sieve algorithm [47]. This attack, published in 2016, asks to update key size. From then on, a group of 256-bit prime

[9] In other words, the trace $t = p + 1 - r$ should be small (compared to p and r).

[10] We choose to employ asymmetric pairings to compute homomorphic product of fresh ciphertexts. The use of symmetric pairings would change the computational hardness assumption [2].

[11] The website `https://gitlab.inria.fr/dldb/discretelogd` covets to list discrete logarithms records in finite fields.

order of an elliptic curve does not ensure anymore 128-bit security level. It has consequences in pairing-based cryptography. In this implementation, a curve is chosen in the Barreto-Naerhig family [48]. It has equation $y^2 = x^3+3$ and is defined over a 256-bit prime field with embedding degree 12 (*i.e.* 3072-bit finite field). According to [44, page 20], such a curve achieves 110 bits of security, while according to [45, page 11] this is around 100 bits.

6 Performance

We first discuss a few technicalities of our scheme. In our implementation, we chose to restrict plaintext space $\mathcal{P}$ to $\mathbb{F}_2$ rather than $\mathbb{Z}_n$ with $n > 2$. Decryption time depends polynomially on the size of $\mathcal{P}$. Moreover, conditional tests *if-then-else* on ciphertexts, which are basic treatments on data, facilitated if encrypted bitwise. On $\mathbb{F}_2$, it is unnecessary to implement the multiplication between a plaintext and a ciphertext (indeed $\mathrm{Enc}(0) \boxplus \mathrm{Enc}(a) = 1 \times \mathrm{Enc}(a)$ and $\mathrm{Enc}(0) = 0 \times \mathrm{Enc}(a)$ where $\mathrm{Enc}(0)$ and $\mathrm{Enc}(a)$ are ciphertext of the same level).

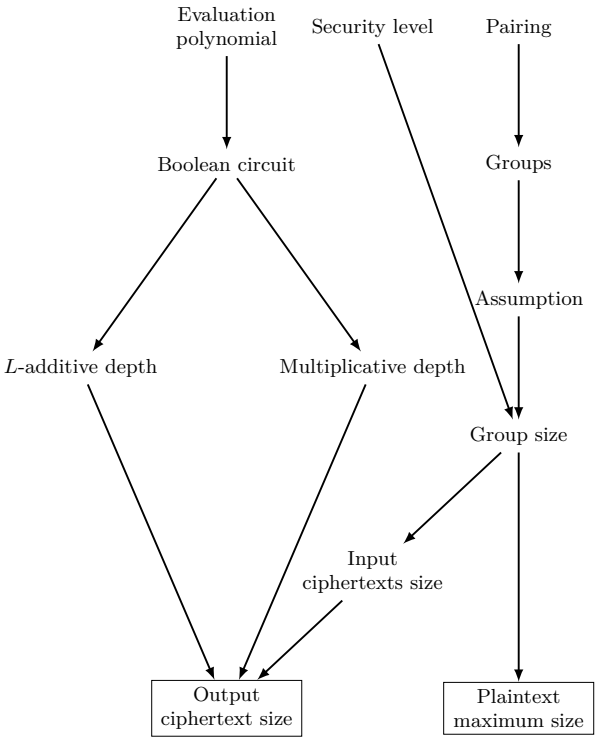

Fig. 2 Parameters influencing memory usage in BGN-F-CF.

We choose to employ optimal Ate pairing [39] over Barreto-Naerhig curves which offers good time performance and is available in library. Note that on a computer algebra system with our parameters (256-bit prime field), best choice, at this security level, in terms of time is the optimal twisted Ate pairing, according to [41]. Our implementation is in software and employs the DCLXVI pairing library (version 20130329) [37]. Originally, the library is optimized for speed performance, not memory usage, and it employs a parameter set which targets 124 or 128-bits security level depending on estimations [49,50]. Another choice could have been Herumi library [51,52], which is the fastest library to compute Optimal Ate pairing over Barreto-Naerhig curve. The DCLXVI library is one-bit more secure than the latter according to its authors. We decided to reuse the library in our implementation of BGN-F-CF scheme.

Figure 2 shows how data size in BGN-F-CF is dependent on two kind of parameters. Those related to functionality (treatments operated on ciphertexts) and those linked with security (implementation choices on cryptosystem settings). Time/space/communication costs depend on:

- operations (encryption scheme, level of ciphertexts)
- operand size (security assumption, security level)
- operation number (Boolean circuit)

Homomorphic operations can differ according to the ciphertext level, as indicated in Section 3. A fresh ciphertext has level 1. Multiplying level L_1 and level L_2 ciphertexts gives a level $L_1 + L_2$ ciphertext.

Ciphertexts sizes are indicated in Table 6, and keys sizes can be found in Table 9. Sizes depend on group order and data representation. In our implementation, we employ 256-bits prime p and r (see Subsection 4) to define groups.

In Table 6, we also give the lower bound minimum size with such p and r. Our implementation uses the DCLXVI library which favours speed rather than memory, this explains the difference between sizes. To estimate minimum size, we use Table 3 and the following reasoning. Ciphertext spaces are composed of elements of $\mathbb{F}_2$, $E(\mathbb{F}_p)$, $E[r] \cap \mathrm{Ker}(\pi - p)$ and μ_r. On our computers, the elementary unit of storage is the byte. We store an element of $\mathbb{F}_2$ on one byte. A point of the group $E(\mathbb{F}_p)$ can be represented by its x-coordinate over $\mathbb{F}_p$ and the sign of its y-coordinate. We can store it with at least $\mathrm{size}(p) + 1$ bytes. A point of the group $E[r] \cap \mathrm{Ker}(\pi - p)$ has coordinates in $\mathbb{F}_{p^{12}}$. Barreto-Naerhig curves have a twist of degree 6. We represent a point of $E[r] \cap \mathrm{Ker}(\pi - p)$ on the twist over $\mathbb{F}_{p^2}$ by storing the x-coordinate in $\mathbb{F}_{p^2}$ and the y-sign. Thus, we can store the point with at least $2 \times \mathrm{size}(p) + 2$ bytes. In the end, an element of the cyclic group μ_r requires at least $\mathrm{size}(r)$ bytes.

Level	Size	Minimum Size
C	784	515
T	1552	1027
2	4624	1025
3	$10032 + A \times 5376$	$2562 + A \times 1538$
4	$13872 + A \times 9216$	$3072 + A \times 2048$

Table 6 Ciphertext size in our BGN-F-CF implementation and minimum size. The unit is the byte. A is the L-additive depth of the circuit which produces a level L ciphertext. The notations C and T refer to a level 1 ciphertext in curve mode and twist mode respectively.

In some configurations, few errors (such as false positives in a test) can be acceptable. In this case, we can restrain constraints and design an ad-hoc circuit with a lower depth than an error-free circuit. Cheap tasks could be done with precomputation and postcomputation. It can permit to decrease both the multiplicative depth and the L-additive depth of the circuit. In BGN-F-CF, the L-additive depth is critical only for $l = 3$ and $l = 4$.

We provide, in Table 7, the running time of different homomorphic operations. Addition of ciphertexts of level > 2 and any multiplication of ciphertexts modify ciphertext spaces (see Table 4). It is thus necessary to specify time for each ciphertext level. All our experiments have been performed with 2 processors Intel Xeon E5-2640 v4, each of them has 10 cores and can manage 20 threads, at a frequency of 2.4 GHz (3.4 GHz in Turbo mode)

To conclude this section, we provide comparison of BGN-F-CF with lattice based solutions mentioned in the introduction. First of all, such a comparison seems natural since today different implementations of such solutions are available. But, we point out that it is not so trivial to perform such comparisons fairly, as these implementations do not always benefit from the same optimizations, and also because some enable to handle plaintext batching while some others do not. We refer the reader to [53] for a more complete list of issues related with implementation and comparison of lattice based solutions. Nevertheless, we did our best to provide the most fair comparison we could. First, we had to select the schemes to compare among the available implementations. Second, we decided to compare the selected schemes for a given security level of about 100-bits (to fit BGN-F-CF security level).

We decided not to include FNTRU [25] nor YASHE [21] (used in SEAL1.0) as they present some security issues [54,55]. We also did not compare with [20] nor its variants [24,26], as they are not particularly interesting in this kind of setting as the startup cost is high [56]. Now, if we focus on available implementations, three remain: HElib[12], SEAL2.1[13], and FV-NFLlib[14]. It is known that HElib efficiency is much better for circuits with higher (typically more than 10) multiplicative depth than for circuits with small multiplicative depth, in which case it is outperformed by FV-NFLlib [57]. Hence, we selected SEAL2.1 and FV-NFLlib, both implement the Fan-Vercauteren scheme [19]. We tuned the parameters of these two libraries to fit the same security level as BGN-F-CF, with the most up-to-date information concerning the security evaluation of both schemes. For BGN-F-CF, we took into account all recent results already presented in Section 5. For lattice based schemes, we used Albrecht's estimator (available on BitBucket[15]), which is the best available security estimator today for such schemes. We provide in Appendix B more detailed information about the settings we chose for the experiments.

We provide in Table 8 running times for elementary operations with SEAL2.1 and FV-NFLlib, and in Tables 9 and 10 some comparison of BGN-F-CF with SEAL2.1

Operation	Level	Time in ms
Encryption	C	0.57
	T	0.40
Multiplication	(C, T)	5.27
	$(C, 2)$	3.02
	$(T, 2)$	2.45
	$(2, 2)$	2.02
Addition	(C, C)	0.28
	(T, T)	0.22
	$(2, 2)$	1.47
	$(3, 3)$	1.36
	$(4, 4)$	1.12
Decryption	C	0.70
	T	0.56
	2	5.03
	3	9.26
	4	16.66

Table 7 Running time for operations in BGN-F-CF scheme. As in BGN-F-CF ciphertexts may belong to different levels, we provide detailed running times with respect to the levels. We consider $A = 0$, see Table 4. The notations C and T refer to a level 1 ciphertext in curve mode and twist mode respectively.

[12] `https://github.com/shaih/HElib`

[13] `https://sealcrypto.codeplex.com/`

[14] `https://github.com/CryptoExperts/FV-NFLlib`

[15] `https://bitbucket.org/malb/lwe-estimator`, commit `eb45a74`

Implementation	Operation	Time in ms
SEAL-2.1	Addition	0.22
	Multiplication	71.84
FV-NFLlib	Addition	0.03
	Multiplication	4.93

Table 8 Single homomorphic operation running times for the selected lattice-based schemes used in our benchmark. Used settings are given in Appendix B.

and FV-NFLlib in terms of data size and running times for a whole circuit processing. In Table 9, we indicate a ciphertext size of 2336 bytes, for BGN-F-CF. This is the worst case, the one where the evaluated circuit needs both, the encryption in curve mode (784 bytes) and in twist mode (1552 bytes). Ciphertext size is particularly important for the client who encrypts data before transmitting them to the server. The server performs homomorphic computations over received ciphertexts to obtain the *evaltext*. We call *evaltext*, the ciphertext obtained after homomorphically evaluating a specified circuit over input ciphertexts.

Implementation	Size in bytes				
	pk	**ek**	**sk**	Ciphertext	Evaltext
BGN2	4608	N/A	1024	2336	15408
SEAL-2.1	32784	65568	16392	30735	76837
FV-NFLlib	28672	24576	8192	12288	12288

Table 9 Implementations comparison overview in terms of data size on a whole test circuit (which is given in Appendix A.3. The lattice-based schemes introduce a third key, the evaluation-key (also called relinearisation-key), denoted **ek**.

Implementation	Time in ms/byte			
	Key-gen	Enc	Eval	Dec
BGN2	0.95	1.20	6.62	9.26
SEAL-2.1	167.53	9.86	304.66	13.23
FV-NFLlib	2.82	2.42	17.23	2.32

Table 10 Implementations comparison running time overview on a whole test circuit (described in Section A.3). Note that, encryption time takes into account encoding time.

The most demanding challenge in homomorphic cryptography is to lower the cost of evaluation. From the results presented above we see that BGN-F-CF is one of the best performing schemes for low multiplicative depth circuits, specially for evaluation. Moreover, it has the added advantage not to need plaintext encoding, nor evaluation key.

We have to mention that, very recently, a new version of SEAL has been released, namely SEAL2.2. As it is around five to six times faster than SEAL2.1, it is evident from our experimental results that this improvement has little impact upon the conclusion of the present work.

We note, BGN-F-CF has smaller keys and no evaluation key to manage. It has also no noise parameter to deal with for correct decryption. In addition, it offers better homomorphic evaluation time and well understood security.

The full implementation code of our scheme, BGN2 is available as "proof of concept" at:

`https://github.com/BGN2/BGN2`.

7 Conclusion

In this paper, we proposed a variant of BGN homomorphic encryption scheme that is called BGN-F-CF and can address one more multiplicative depth. This scheme may help to address practical situations where the multiplicative depth is of 2, with smaller keys and ciphertexts than homomorphic encryption schemes based on lattices. Moreover, its security is better understood than for lattice based schemes, as it relies on Discrete Logarithm computation, which has been studied more extensively than, for example, LWE or RLWE. It is also less complex, as no bootstrapping nor relinearization is needed here.

Furthermore, it is worth noting that, while the lattice based schemes seem to have more potential, the security and parameter choices for them remain to be open problems. On the other hand the security of our scheme is well understood and alternative systems are always of interest!

Acknowledgment

We deeply thank Aurore Guillevic for our discussions concerning the security estimation of our scheme according to the last attacks published. This work has been funded by Region Bretagne under grant AAP PME 2014, 14006192.

References

1. Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF Formulas on Ciphertexts. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2005.
2. David Mandell Freeman. Converting Pairing-Based Cryptosystems from Composite-Order Groups to Prime-Order Groups. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 44–61. Springer, 2010.
3. Dario Catalano and Dario Fiore. Using Linearly-Homomorphic Encryption to Evaluate Degree-2 Functions on Encrypted Data. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 1518–1529. ACM, 2015.
4. Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Proc. of Advances in Cryptology — EUROCRYPT 1999*, number 1592 in LNCS, pages 223–238, 1999.
5. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
6. Dan Boneh. The Decision Diffie-Hellman Problem. In *Proceedings of the Third International Symposium on Algorithmic Number Theory*, ANTS-III, pages 48–63, London, UK, UK, 1998. Springer-Verlag.
7. Nadia El Mrabet and Marc Joye. *Guide to Pairing-Based Cryptography*. CRC Press, 2017.
8. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.
9. Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.
10. Tiziano Bianchi, Alessandro Piva, and Mauro Barni. On the implementation of the discrete Fourier transform in the encrypted domain. *IEEE Transactions on Information Forensics and Security*, 4(1):86–97, 2009.
11. Carlos Aguilar-Melchor, Philippe Gaborit, and Javier Herranz. Additively homomorphic encryption with d-operand multiplications. In *Advances in Cryptology–CRYPTO 2010*, pages 138–154. Springer, 2010.
12. Kristin Lauter, Adriana López-Alt, and Michael Naehrig. Private computation on encrypted genomic data. In *International Conference on Cryptology and Information Security in Latin America*, pages 3–27. Springer, 2014.
13. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in cryptology–EUROCRYPT 2010*, pages 24–43. Springer, 2010.
14. Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *International Workshop on Public Key Cryptography*, pages 420–443. Springer, 2010.
15. Michael Naehrig, Kristin E. Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *ACM CCSW*, pages 113–124. ACM, 2011.
16. Craig Gentry, Shai Halevi, and Nigel P Smart. Fully homomorphic encryption with polylog overhead. In *Advances in Cryptology–EUROCRYPT 2012*, pages 465–482. Springer, 2012.
17. Craig Gentry and Shai Halevi. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 107–109. IEEE, 2011.
18. Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. Public key compression and modulus switching for fully homomorphic encryption over the integers. In *Advances in Cryptology–EUROCRYPT 2012*, pages 446–464. Springer, 2012.
19. Junfeng Fan and Frederik Vercauteren. Somewhat Practical Fully Homomorphic Encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.
20. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology–CRYPTO 2013*, pages 75–92. Springer, 2013.
21. Joppe W Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme. In *International Conference On Cryptography and Coding*. Springer Verlag, December 2013.
22. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) Fully Homomorphic Encryption without Bootstrapping. *TOCT*, 6(3):13, 2014.
23. Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE As Secure As PKE. In *Proc. of the 5th Conference on Innovations in Theoretical Computer Science – ITCS 2014*, pages 1–12. ACM, 2014.
24. Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers. *IEEE Transactions on Computers*, PP(99):1–1, 2015.
25. Y. Doröz and B. Sunar. Flattening NTRU for Evaluation Key Free Homomorphic Encryption. Cryptology ePrint Archive, Report 2016/315, 2016.
26. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, pages 3–33, 2016.
27. Frederik Armknecht and Ahmad-Reza Sadeghi. A New Approach for Algebraically Homomorphic Encryption. *IACR Cryptology ePrint Archive*, 2008:422, 2008.
28. Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 28–37. IEEE, 1998.
29. Daniel Augot and Matthieu Finiasz. A public key encryption scheme based on the polynomial reconstruction problem. *Advances in Cryptology—EUROCRYPT 2003*, pages 645–645, 2003.
30. Aggelos Kiayias and Moti Yung. Directions in polynomial reconstruction based cryptography. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 87(5):978–985, 2004.
31. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.
32. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. *Advances in Cryptology – EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 – June 3, 2010. Proceedings*, chapter On Ideal Lattices and Learning with Errors over Rings, pages 1–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

33. Aurore Guillevic. *Arithmetic of pairings on algebraic curves for cryptography*. Theses, Ecole Normale Supérieure de Paris - ENS Paris, December 2013.
34. Aurore Guillevic. Comparing the Pairing Efficiency over Composite-Order and Prime-Order Elliptic Curves. In *Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings*, pages 357–372, 2013.
35. David Freeman, Michael Scott, and Edlyn Teske. A Taxonomy of Pairing-Friendly Elliptic Curves. *Journal of Cryptology*, 23(2):224–280, 2009.
36. Kim Laine, Hao Chen, and Rachel Player. Simple Encrypted Arithmetic Library - SEAL (v2.1). Technical report, September 2016.
37. Michael Naehrig, Ruben Niederhagen, and Peter Schwabe. New Software Speed Records for Cryptographic Pairings. In Michel Abdalla and Paulo S. L. M. Barreto, editors, *Progress in Cryptology - LATINCRYPT 2010, First International Conference on Cryptology and Information Security in Latin America, Puebla, Mexico, August 8-11, 2010, Proceedings*, volume 6212 of *Lecture Notes in Computer Science*, pages 109–123. Springer, 2010.
38. Steven Galbraith, Florian Hess, and Frederik Vercauteren. Hyperelliptic pairings. *Pairing-Based Cryptography–Pairing 2007*, pages 108–131, 2007.
39. Frederik Vercauteren. Optimal pairings. *IEEE Transactions on Information Theory*, 56(1):455–461, 2010.
40. Laura Hitt. On the minimal embedding field. *Pairing-Based Cryptography–Pairing 2007*, pages 294–301, 2007.
41. Andreas Enge and Jérôme Milan. *Security, Privacy, and Applied Cryptography Engineering: 4th International Conference, SPACE 2014, Pune, India, October 18-22, 2014. Proceedings*, chapter Implementing Cryptographic Pairings at Standard Security Levels, pages 28–46. Springer International Publishing, Cham, 2014.
42. Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In *Annual Cryptology Conference*, pages 543–571. Springer, 2016.
43. Sylvain Duquesne, Nadia El Mrabet, Safia Haloui, and Franck Rondepierre. Choosing and generating parameters for low level pairing implementation on BN curves. 2015.
44. Alfred Menezes, Palash Sarkar, and Shashank Singh. Challenges with Assessing the Impact of NFS Advances on the Security of Pairing-based Cryptography. In *Proceedings of Mycrypt*, 2016.
45. Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. Cryptology ePrint Archive, Report 2017/334, 2017. `http://eprint.iacr.org/2017/334`.
46. Ezekiel J Kachisa, Edward F Schaefer, and Michael Scott. Constructing brezing-weng pairing-friendly elliptic curves using elements in the cyclotomic field. *Pairing*, 8:126–135, 2008.
47. Arjen K Lenstra, Hendrik W Lenstra Jr, Mark S Manasse, and John M Pollard. The number field sieve. In *The development of the number field sieve*, pages 11–42. Springer, 1993.
48. Paulo S. L. M. Barreto and Michael Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. In Bart Preneel and Stafford E. Tavares, editors, *Selected Areas in Cryptography, 12th International Workshop, SAC 2005, Kingston, ON, Canada, August 11-12, 2005, Revised Selected Papers*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2005.
49. Barker Elaine, Barker William, Burr William, Polk William, and Smid Miles. Recommendation for key management–part 1: general. *NIST Special Publication*, pages 800–57, 2006.
50. Nigel Smart et al. Ecrypt2 yearly report on algorithms and keysizes (2008-2009). Technical report, Technical report, ECRYPT II–European Network of Excellence in Cryptology, EU FP7, ICT-2007-216676, 2009. published as deliverable D. SPA. 7 http://www. ecrypt. eu. org/documents/D. SPA. 7. pdf, 2009.
51. Shigeo Mitsunari. A Fast Implementation of the Optimal Ate Pairing over BN curve on Intel Haswell Processor. *IACR Cryptology ePrint Archive*, 2013:362, 2013.
52. Jean-Luc Beuchat, Jorge E. González-Díaz, Shigeo Mitsunari, Eiji Okamoto, Francisco Rodríguez-Henríquez, and Tadanori Teruya. *High-Speed Software Implementation of the Optimal Ate Pairing over Barreto–Naehrig Curves*, pages 21–39. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
53. Guillaume Bonnoron, Caroline Fontaine, Guy Gogniat, Vincent Herbert, Vianney Lapôtre, Vincent Migliore, and Adeline Roux-Langlois. *Somewhat/Fully Homomorphic Encryption: Implementation Progresses and Challenges*, pages 68–82. Springer International Publishing, Cham, 2017.
54. Martin Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions: Cryptanalysis of some FHE and Graded Encoding Schemes. Cryptology ePrint Archive, Report 2016/127, 2016.
55. Paul Kirchner and Pierre-Alain Fouque. Comparison between Subfield and Straightforward Attacks on NTRU. Cryptology ePrint Archive, 2016/717, 2016.
56. Vincent Migliore, Guillaume Bonnoron, and Caroline Fontaine. Determination and exploration of practical parameters for the latest Somewhat Homomorphic Encryption (SHE) Schemes. working paper or preprint, October 2016.
57. Carlos Aguilar-Melchor. Private communication.

A Low multiplicative depth Boolean circuits

We treat bit per bit encryption. Circuits are rewritten with two operators: $\veebar$ (exclusive disjunction) and $\wedge$. They can be written under different forms depending on operation order.

A.1 Binary data

Table 11 supplies constraints on ciphertext levels with typical examples of low-depth multiplicative circuits.

Notation: ciphertexts a (resp. b, c, d) with level L_1 (resp. level L_2, L_3, L_4), ciphertexts x (resp. y) with level M_1 (resp. level M_2).

In BGN-F-CF, no multiplication is defined with a factor having a level ≥ 3. Therefore, operands level is limited according to multiplicative depth.

[16] A Boolean function is not associated to a unique Boolean circuit. We restrict to circuits with $\veebar$ and $\wedge$ gates because these gates correspond to elementary homomorphic operations in BGN-F-CF. The parentheses indicates the order of operations. It permits to define a unique Boolean circuit and then to indicate its multiplicative depth.

Functions	Boolean function[16]	Mult. Depth	Input Level	Output Level
Test $a == 1$	a	0	$[\![1,4]\!]$	L_1
REFRESH a	$a \veebar 0$			
Test $a == 0$, NOT a	$a \veebar 1$			
Test $a \neq b$, a XOR b	$(a \veebar b)$			$\max(L_1, L_2)$
Test $a == b$, a XNOR b	$(a \veebar b) \veebar 1$			
a AND b	$a \wedge b$	1	$[\![1,2]\!]$	$L_1 + L_2$
a OR b	$((a \wedge b) \veebar a) \veebar b$			
2-to-1 MUX $(x\ ?\ a : b)$	$((a \veebar b) \wedge x) \veebar b$			$\max(L_1, L_2) + M_1$
4-to-1 MUX inputs: x, y output: $\in \{a, b, c, d\}$	$\big((a \wedge x \wedge y) \veebar (b \wedge x \wedge (y \veebar 1))\big) \veebar \big((c \wedge (x \veebar 1) \wedge y) \veebar (d \wedge (x \veebar 1) \wedge (y \veebar 1))\big)$	1	1	$\max(L_1, L_2, L_3, L_4) + M_1 + M_2$

Table 11 Ciphertext levels with some low-depth multiplicative circuits. Notations are given in Section A.1.

A.2 Integer data

Data are n-bits integers. Input ciphertexts have level $L = 1$.

- Adder $a_{n-1} \dots a_0 + b_{n-1} \dots b_0 \mod 2^n$
  In terms of multiplicative depth, the hardest part is the evaluation of the carry which enables to compute the most significant bit (MSB) of the sum modulo 2^n. Let us compute the MSB with $n = 3$, which is the maximal value for BGN-F-CF. Then, it can be written:

  $$a_2 \veebar b_2 \veebar ((a_1 \wedge b_1) \wedge (a_0 \wedge b_0)).$$

  The Boolean circuit has multiplicative depth $\lceil \log_2 2(n-1) \rceil = 2$. The corresponding ciphertext is the sum of three ciphertexts of level $2(n-1)L = 4$. Indeed, to add ciphertexts, we need to have ciphertexts of the same level. In this case a_2 and b_2 have level L but $((a_1 \wedge b_1) \wedge (a_0 \wedge b_0))$ has level 4. To increment the level of a ciphertext, we multiply it homomorphically by an encryption of 1.
- Test $a_{n-1} \dots a_0 == b_{n-1} \dots b_0$.
  With BGN-F-CF, we can manage up to $n = 4$ bits with the circuit of depth $\lceil log_2(n) \rceil = 2$:

  $$((a_3 \veebar b_3) \veebar 1) \wedge ((a_2 \veebar b_2) \veebar 1) \wedge ((a_1 \veebar b_1) \veebar 1) \wedge ((a_0 \veebar b_0) \veebar 1).$$

  The output is a ciphertext of level $nL = 4$.

A.3 Evaluation circuit

We compare SEAL 2.1, FV-NFLlib and BGN-F-CF on a 2-depth test circuit, the results are given in Table 9 and Table 10. The homomorphic circuit is a toy example. It takes as input an encrypted letter and homomorphically changes lower-case letter into upper-case letter. Precomputation and postcomputation are done. Generally, they can be solutions to decrease the multiplicative depth of the homomorphic circuit. In this example, the homomorphic circuit tests characterwise if the character is a lower-case letter or not.
Let the input character, encoded in extended ASCII, be written as λ. We denote the bits of $\lambda+31$ as $n_9n_8n_7n_6n_5n_4n_3n_2n_1$ and the bits of $\lambda + 5$, as $o_9o_8o_7o_6o_5o_4o_3o_2o_1$. We precompute n_8, n_9 and o_8. On plaintexts, the lower-case test consists in the Boolean expression:

$$n_8 \wedge \neg n_9 \wedge \neg o_8$$

We evaluate a corresponding homomorphic circuit:

$$\mathrm{Enc}(n_8) \boxtimes (\mathrm{Enc}(\mathbf{1}) \boxplus \mathrm{Enc}(n_9)) \boxtimes (\mathrm{Enc}(\mathbf{1}) \boxplus \mathrm{Enc}(o_8))$$

The results for this circuit gives the insight of how the schemes compare with each other. The trend should remain similar for other circuits. Note that precomputation and postcomputation time are taken into account in Table 9 and Table 10.

B Parameter choices of SEAL 2.1 and FV-NFLlib

We compare our construction of BGN-F-CF with SEAL 2.1 and FV-NFLlib which are implementations of the Fan-Vercauteren scheme[19]. It is worth noting that the parameter choices and their effect on security and performance of both the lattice based schemes, SEAL 2.1 and FV-NFLlib, are not yet definitive. We compared the schemes for a given *security level*.
The tool we used for security estimation of the lattice based schemes, by Martin Albrecht and is available at `https://bitbucket.org/malb/lwe-estimator`, commit `eb45a74`. Note, this is an upper level security estimator not considering the dedicated attacks on definitive schemes. We compare the three schemes for a given security level of about 100-bits.
Let n is the *plaintext modulus*, q be the *coefficient modulus* and σ is the *standard deviation of the Gaussian noise* (terms used from [36]). Regev gave a lemma [31] to define the noise parameter α, which can be expressed as:

$$\alpha = \frac{\sqrt{2\pi}\sigma}{q}.$$

- $n = 2048$,
- $q = 2^{74} - 2^{14} + 1$,
- $\sigma = 65$.