Improved Filter Permutators for Efficient FHE: Better Instances and Implementations

Pierrick Méaux[1], Claude Carlet[2],
Anthony Journault[1], François-Xavier Standaert[1].

[1] ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium.
[2] LAGA, University of Paris 8, France and Dep. of Informatics, University of Bergen, Norway.

Abstract. We revisit the design of filter permutators as a general approach to build stream ciphers that can be efficiently evaluated in a fully homomorphic manner. We first introduce improved filter permutators that allow better security analyses, instances and implementations than the previously proposed FLIP family of ciphers. We also put forward the similarities between these improved constructions and a popular PRG design by Goldreich. We then propose a methodology to evaluate the performance of such symmetric cipher designs in a FHE setting, which primarily focuses on the noise level of the symmetric ciphertexts (hence on the amount of operations on these ciphertexts that can be homomorphically evaluated). Evaluations through HElib show that instances of improved filter permutators using direct sums of monomials as filter outperform all existing ciphers in the literature based on this criteria. We also discuss the (limited) overheads of these instances in terms of latency and throughput.
keywords: Filter Permutator, Homomorphic Encryption, Boolean Functions.

1 Introduction.

State-of-the-art. Block cipher designs with reduced multiplicative complexity (*e.g.* number of AND gates per ciphertext bit or AND depth) have recently attracted significant attention in symmetric cryptography research. Such ciphers are motivated by new constraints raised by emerging security applications. For example, limited multiplicative complexity allows preventing side-channel attacks via masking more efficiently [24,27,40], can improve the throughput and latency of Multi-Party Computation (MPC) protocols [2, 26], and mitigates the noise increase and the ciphertext expansion in Fully Homomorphic Encryption (FHE) schemes [2, 3, 10, 19, 36]. Concretely, thanks to innovative (and sometimes aggressive) design choices, recent ciphers (*e.g.* LowMC [2, 3]) can encrypt with as little of four ANDs per bit, or with a multiplicative depth of four (*e.g.* FLIP [36]). In a recent work by Dobraunig et al., the authors even go as far as minimizing both metrics jointly for a single cipher (called Rasta [19]). In this paper, we are concerned with the question whether the reduction of the multiplicative depth can be pushed even further (yet, with "reasonable" key sizes).

More specifically, we are interested in the exploitation of Symmetric Encryption for FHE applications (SE-FHE), which we will sometimes call "hybrid FHE framework", and in stream ciphers based on Filter Permutators (FPs) introduced at Eurocrypt 2016 by Méaux et al. [36]. While the simple structure of FPs is particularly appealing for

FHE, early instances of the FLIP cipher[1] have been cryptanalyzed by Duval et al. thanks to guess-and-determine attacks [20]. These attacks led the authors of FLIP to tweak the designs published at Eurocrypt with conservative choices (hence reducing their performances). They also highlighted a lack of theoretical understanding of the exact properties needed for the Boolean functions used in FPs, due to the fact that these designs exploit Boolean functions with non-uniformly distributed inputs. As later observed by Carlet et al. such a structure leads to new research problems related to the analysis of Boolean functions [12, 34, 37, 38]. Besides, recent results confirmed that the design principles of FPs are at least theoretically sound (thanks to an analysis in the random oracle model) [13], hence leaving ample room for improved constructions and analyzes of stream ciphers taking advantage of the FP principles in order to enable efficient SE-FHE.[2]

Contributions. Building on this state-of-the-art, our first contribution is to propose a new family of stream ciphers, namely Improved Filter Permutators (IFPs) that takes advantage of recent advances on FPs and the cipher FLIP. It mostly tweaks FPs in two directions. First, IFPs exploit an extended key register, so that the key size N is larger than the input of the Boolean function used as a filter. This allows the input of the filter to be increasingly close to uniformly distributed as N increases (which is not the case for FPs). Second, IFPs use a whitening stage. That is, before being sent to the filter the permuted key bits are XORed with a random public value. This allows simplifying the analysis of guess-and-determine attacks (and to mitigate them), while having no impact on the noise increase and a very mild impact on the latency and throughput of a SE-FHE implementation. The register extension also makes the IFP designs more similar to a PRG construction proposed by Goldreich [25]. So despite most known results on Goldreich's PRG are asymptotic [5, 6, 18, 39], they also give confidence that the general principles of IFPs are sound and motivate the quest for efficient instances.

We provide a detailed security analysis of IFPs (considering all published attacks we are aware of) and use it to propose new cipher instances exploiting Boolean filters made of Direct Sums of Monomials (DSMs). We denote these new instances as $\mathsf{FiLIP}_{\mathsf{DSM}}$ and observe that they allow reduced multiplication complexity compared to existing instances of FLIP. The security analysis largely depends on cryptographic criteria of Boolean functions, and more particularly on the properties of sub-functions derived from the filtering function. Therefore we develop the tools required for this analysis, and we exhibit the parameters of any DSM function relatively to these relevant criteria.

We then use these new instances as a basis to compare IFPs with other published ciphers aimed at efficient FHE evaluation. In this respect, we first observe that the usual approach for assessing such performances is to compare latency and throughput [2, 10, 19, 36]. The best performances are then typically achieved for FHE parameters such that the ciphertexts will "just enable decryption", or a few levels of multiplications more. Directly optimizing the latency and throughput of the obtained homomorphic ciphertexts forces to fix the ciphertext size and error. On the one hand, this methodology

[1] More precisely, instances presented at the "Journées C2", a French workshop for PhD students held in La Londe Les Maures, in October 2015.

[2] To some extent, the Rasta cipher actually exploits one of the FPs' ideas, which is to make a part of the cipher computations independent of the key.

gives accurate timings relatively to the targeted function, or close ones. On the other hand, it makes the evaluations quite application-specific since the estimation of these metrics is quite dependent of the targeted function. More precisely, this optimization leads to tailor the homomorphic ciphertext parameters for the SE evaluation. Thus, the ciphertexts have different sizes, so involving different times to evaluate the same function, and different quantities of noise which lead to different security levels. We therefore argue that in order to evaluate various symmetric schemes for application in a hybrid-FHE framework in an application-dependent manner, a more relevant comparison metric is based on the noise level of the ciphertexts. This leads to a more stable metric than latency and throughput (since it avoids the aforementioned specialization to a given target function). It is also connected to a generally desirable goal (since one may expect that the ability to perform as much homomorphic operations as possible is a useful feature for practical applications).

We formalize our comparison methodology, by (*i*) setting the FHE security parameters at a level that is comparable to the SE ones (*i.e.* 80-bit or 128-bit), (*ii*) using ciphertexts of comparable size for all the cipher designs to compare (so basing the comparison on the most expensive cipher in terms of noise), and (*iii*) monitoring the noise (*e.g.* provided by HElib) not only for the ciphertexts but also after one and two levels of additional multiplications on the ciphertexts. Concrete estimations carried out using HElib put forward that the noise of Rasta and FiLIP is orders of magnitudes smaller than the one of LowMC, and that new instances of FiLIP with reduced multiplicative depth allow performing two more levels of multiplications on its ciphertexts than the recommended Rasta designs. We further observe that even non-recommended versions of Rasta (with comparable multiplicative depth) would not compare favorably to FiLIP due to a (much) larger key size. We complement our analyzes with an evaluation of best-case latency and throughput (*i.e.* when ciphertexts can just be decrypted), as performed previously. We believe that it remains an informative alternative metric and clarify that it has to be understood as the best possible performances of a cipher since any concrete application (where symmetric ciphertexts are manipulated homomorphically) will require ciphertext expansion.

2 Preliminaries.

In addition to classic notation we use the $\log$ to denote the logarithm in basis 2, and $[n]$ to denote the subset of all integers between 1 and n: $\{1, \ldots, n\}$. For readability we use the notation $+$ instead of $\oplus$ to denote the addition in $\mathbb{F}_2$.

2.1 Boolean Functions and Criteria.

We introduce here some core notions of Boolean functions in cryptography, restricting our study to the following definition of Boolean function, more restrictive than a vectorial Boolean function. We recall the main cryptographic properties of Boolean functions, mostly taken from [11]: balancedness, resiliency, nonlinearity and (fast) algebraic immunity. We give notions relatively to bit-fixing (as defined in [6]) due to there important when guess-and-determine attacks are investigated (see Section 4.3). Finally we define the direct sums of monomials and recall some of their properties.

A Boolean function f with n variables is a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$. The set of all Boolean functions in n variables is denoted by $\mathcal{B}_n$. We call Algebraic Normal Form

(ANF) of a Boolean function f its n-variable polynomial representation over $\mathbb{F}_2$ (*i.e.* belonging to $\mathbb{F}_2[x_1,\dots,x_n]/(x_1^2+x_1,\dots,x_n^2+x_n)$):

$$f(x) = \sum_{I\subseteq[n]} a_I \left(\prod_{i\in I} x_i\right) = \sum_{I\subseteq[n]} a_I x^I,$$

where $a_I \in \mathbb{F}_2$. The algebraic degree of f equals the global degree $\max_{\{I \mid a_I=1\}} |I|$ of its ANF. Any term $\prod_{i\in I} x_i$ is called a monomial and its degree equals $|I|$. A function with only one non-zero coefficient a_I, $|I|>0$, is called a monomial function.

A Boolean function $f \in \mathcal{B}_n$ is said to be balanced if its output is uniformly distributed over $\{0,1\}$. f is called m-resilient if any of its restrictions obtained by fixing at most m of its coordinates is balanced. We denote by $\mathsf{res}(f)$ the maximum resiliency m of f and set $\mathsf{res}(f) = -1$ if f is unbalanced. The nonlinearity NL of a Boolean function f is the minimum Hamming distance between f and all the affine functions in $\mathcal{B}_n$: $\mathsf{NL}(f) = \min_{g,\,\mathsf{deg}(g)\leq 1}\{d_H(f,g)\}$, with $d_H(f,g) = \#\{x\in\mathbb{F}_2^n \mid f(x)\neq g(x)\}$ the Hamming distance between f and g; and $g(x) = a\cdot x+\varepsilon$, $a\in\mathbb{F}_2^n, \varepsilon\in\mathbb{F}_2$ (where $\cdot$ is some inner product in $\mathbb{F}_2^n$; any choice of an inner product will give the same definition). The algebraic immunity of a Boolean function, denoted as $\mathsf{AI}(f)$, is defined as:

$$\mathsf{AI}(f) = \min_{g\neq 0}\{\mathsf{deg}(g) \mid fg = 0 \text{ or } (f+1)g = 0\},$$

where $\mathsf{deg}(g)$ is the algebraic degree of g. The function g is called an annihilator of f (or $f+1$). We additively use the notation $\mathsf{AN}(f)$ for the minimum algebraic degree of non null annihilator of f, and $\mathcal{D}\mathsf{AN}(f)$ for the dimension of the vector space made of the annihilators of f of degree $\mathsf{AI}(f)$ and the zero function. The fast algebraic immunity, denoted as $\mathsf{FAI}(f)$, is defined (*e.g.* [7]) as:

$$\mathsf{FAI}(f) = \min\{2\mathsf{AI}(f), \min_{1\leq\mathsf{deg}(g)<\mathsf{AI}(f)}(\max[\mathsf{deg}(g)+\mathsf{deg}(fg), 3\mathsf{deg}(g)])\}.$$

Definition 1 (Bit-fixing Descendants and Bit-fixing Families). *Let f be a Boolean function in n variables (x_i, for $i\in[n]$), let ℓ be an integer such that $0\leq\ell<n$, let $I\subset[n]$ be of size ℓ (i.e. $I=\{I_1,\dots,I_\ell\}$ with $I_i<I_{i+1}$ for all $i\in[\ell-1]$), and let $b\in\mathbb{F}_2^\ell$, we denote as $f_{I,b}$ the ℓ-bit fixing descendant of f on subset I with binary vector b the Boolean function in $n-\ell$ variables:*

$$f_{I,b}(x') = f(x) \mid \forall i\in[\ell],\ x_{I_i} = b_i, \quad \textit{where } x' = (x_i, \textit{ for } i\in[n]\backslash I).$$

For $\mathcal{F}$ a family of Boolean functions, $\mathcal{F}$ is called bit-fixing stable, or stable relatively to guess and determine, if for all $f\in\mathcal{F}$ all its descendants belong to $\mathcal{F}$.

Definition 2 (Direct Sum of Monomials and Direct Sum Vector). *Let $f\in\mathcal{B}_n$, we call f a Direct Sum of Monomials (or DSM) if the following holds for its ANF:*

$$\forall (I,J) \textit{ such that } a_I = a_J = 1,\ I\cap J\in\{\emptyset, I\cup J\}.$$

We define its Direct Sum Vector (or DSV) $\mathbf{m}_f$ of length $k=\mathsf{deg}(f)$ where m_i ,$i\in[k]$, is the number of monomials of degree i:

$$\mathbf{m}_f = [m_1, m_2, \dots, m_k], \quad m_i = |\{a_I = 1, \textit{ such that } |I| = i\}|.$$

By default when we refer to the vector $\mathbf{m}_F = [m_1, m_2, \dots, m_k]$, it corresponds to the function in $N = \sum_{i=1}^k i m_i$ variables with constant term being null.

Remark 1. Some properties of two particular families of DSM have been studied in [36], the family of triangular functions, *i.e.* $\mathbf{m}_F$ is the all-1 vector of length k, and FLIP functions, *i.e.* $\forall i \in [3,k]\ m_i = m_k$, $m_1 > m_k$, and $m_2 > m_k$.

Note also that DSM functions form a bit-fixing stable family: fixing $\ell < n$ variables (influencing $f(x)$ of not) does not change the property on the ANF defining a DSM.

2.2 Fully Homomorphic Encryption.

We recall here the definition of (fully) homomorphic encryption, a kind of encryption enabling to perform computations on plaintexts only manipulating the ciphertexts, without requiring the ability of decrypting. We introduce the vocabulary relative to homomorphic encryption we will use in this paper. For more details we refer to [22] for FHE, and to [31, 36] for hybrid homomorphic encryption.

Definition 3 (Homomorphic Encryption Scheme). *Let $\mathcal{M}$ be the plaintext space, $\mathcal{C}$ the ciphertext space and λ the security parameter. A homomorphic encryption scheme consists of four probabilistic polynomial-time algorithms:*

- $H.\mathsf{KeyGen}(1^\lambda)$. *Generates a pair $(\mathsf{pk}^H, \mathsf{sk}^H)$, public and secret keys of the scheme.*
- $H.\mathsf{Enc}(m, \mathsf{pk}^H)$. *From the plaintext $m \in \mathcal{M}$ and pk^H, outputs a ciphertext $c \in \mathcal{C}$.*
- $H.\mathsf{Dec}(c, \mathsf{sk}^H)$. *From the ciphertext $c \in \mathcal{C}$ and the secret key, outputs $m' \in \mathcal{M}$.*
- $H.\mathsf{Eval}(f, c_1, \cdots, c_k, \mathsf{pk}^H)$. *With $c_i = H.\mathsf{Enc}(m_i, \mathsf{pk}^H)$ for $1 \leq i \leq k$, outputs a ciphertext $c_f \in \mathcal{C}$.*

Homomorphic encryption: simple, leveled, somewhat, fully. Different notions of homomorphic encryption exist, depending on the set over which the function f can be taken, that is, on the operations which are possible. For all these kinds we assume a compactness property: $|\mathcal{C}|$ is finite, and the size of a ciphertext does not depend on the number of operations performed to obtain it. When only one kind of operation is permitted the scheme is simply homomorphic, it is called somewhat homomorphic when more than one operation can be performed, at least partially. Leveled homomorphic schemes correspond to f being any polynomial of bounded degree (defining the level) and bounded coefficients. Fully Homomorphic Encryption (FHE) corresponds to f being any function defined over $\mathcal{M}$. Gentry [22] proved that FHE can be constructed by combining a leveled homomorphic scheme with a bootstrapping technique. As this technique is still a bottleneck for homomorphic evaluation, we consider a framework where no bootstrapping (or at least less bootstrappings) are performed, and then when we refer to FHE or HE it refers more precisely to this context.

Noise or error-growth. Any known FHE scheme is based on noise-based cryptography, so that an homomorphic ciphertext is associated to a part of error (or noise). The more homomorphic operations are performed, the higher is the noise (if no bootstrapping is used), this quantity of noise can be measured in terms of standard deviation of the distribution followed by the error part. The error-growth involved in an homomorphic evaluation is then the evolution of this parameter.

FHE generations. Since Gentry's breakthrough [22], various FHE schemes following this blueprint appeared. We call second generation the schemes where the error of the product is symmetric in the factors, as BGV [9] which is often considered for efficiency

comparisons as implemented in the HElib library [28]. We call third generation the schemes where the error of the product is asymmetric in the factors, the most recent generation of FHE, initiated with GSW [23].

2.3 Filter Permutators and FLIP Instances.

The Filter Permutator or FP is the general design of stream ciphers introduced in [36], and FLIP is an instance of this design where the filtering function is taken from a sub-family of DSM functions. The main design principle of FPs is to filter a constant key register with a variable (public) bit permutation. More precisely, at each cycle, the key register is (bitwise) permuted with a pseudo-randomly generated permutation, and then a non-linear filtering function is applied to the output of this permuted key register. The general structure of FPs is depicted in the left part of Figure 1. It is composed of three parts: A register where the key is stored, a (bit) permutation generator parametrized by a Pseudo Random Number Generator (PRNG) which is initialized with a public IV, and a filtering function which generates a key-stream.

3 Improved Filter Permutators: a New Design for Better Security and Better Performances.

Two main tweaks are performed on the Filter Permutators blueprint to increase its security and its performances as a SE scheme in the SE-FHE framework. The first goal of these modifications is to generalize the original design, in a way which provides more flexibility to choose the functions used, and the number of variables involved in the computations. The second goal consists in simplifying the security analysis, erasing some particularities of the FP which make the security difficult to evaluate.

3.1 Description.

The design of Improved Filter Permutators (IFPs) deviates from filter permutators blueprint in two ways. First, the size of the key register and the number of variables of the filtering function is not forced to be equal. The IFP key can be longer than the number of inputs of the filtering function (in practice we consider a small factor between both, between 2 and 32). Second, at each clock cycle a whitening of the size of F input's is derived from the PRNG and bitwise XORed with the permuted sub-part of the key.

It gives a new design depicted in the right part of Figure 1, with the following particularities: N is the size of the key register, $n \leq N$ is the number of selected bits from the key register at each clock cycle, and $F \in \mathcal{B}_n$ is the filtering function.

For a security parameter λ, to encrypt $m \leq 2^\lambda$ bits under a secret key $K \in \mathbb{F}_2^N$ (such that $\mathsf{w_H}(K) = N/2$), the public parameters of the PRNG are chosen and then the following process is executed for each key-stream bit s_i (for $i \in [m]$):

- The PRNG is updated, its output determines the subset, the permutation, and the whitening at time i,
- the subset S_i is chosen, as a subset of n elements over N,
- the permutation P_i from n to n elements is chosen,
- the whitening w_i from $\mathbb{F}_2^n$ is chosen,

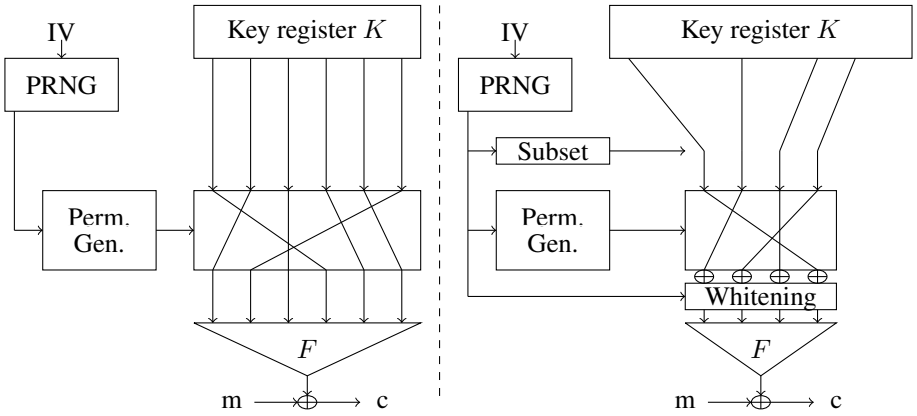

Fig. 1. Filter permutator and improved filter permutator constructions.

– the key-stream bit s_i is computed as $s_i = F(P_i(S_i(K)) + w_i)$, where $+$ denotes the bitwise XOR.

Note that for each clock cycle i we consider that the PRNG gives enough pseudorandom bits to independently determine the subset ($\log \binom{N}{n}$ bits), the permutation ($\log(n!)$ bits), and the whitening (n bits). Its effect on the performances of IFPs in a hybrid FHE framework is negligible anyway. Note that if the number of pseudorandom bits given by the instance of the PRNG used is limited to b, it enables to compute $\lfloor b/(\log \binom{N}{n} + \log(n!) + n) \rfloor$ bits of ciphertexts only. If this quantity is smaller than m, then another instance of PRNG is used, and so forth until the m bits of ciphertexts are produced (an instantiation of the whole scheme is given in Section 5). Any pseudorandom sequence not adversarially chosen could be used instead of the PRNG's output, the use of the PRNG is only motivated by the storage limitation [36] of one of the participants in the hybrid FHE framework.

3.2 Impact on Security.

The two modifications from FPs to IFPs, *i.e.* the register extension and the whitening, are generalizing the design and strictly improving the security. The register extension has two main advantages. First, it enables to increase the security without using more complex functions (allowing then more flexibility in the design). Indeed, keeping invariant the filtering function, increasing N decreases the probability of each key-bit to appear in a key-stream equation, directly increasing the complexity of all attacks known to apply on the Filtering Permutator. Second, the Hamming weight of F's input is not constant anymore. Since $N \geq 2n$, F can be evaluated on any element of $\mathbb{F}_2^n$ (being the weight of $S_i(K)$ at time i), it makes the attacks based on restricted input considerations [12] even less efficient.

The main advantage of the whitening is to facilitate the analysis of security against guess-and-determine attacks [20]. When a guess-and-determine strategy is used by the attacker, some key bits (ℓ) are fixed and then the key-stream bits do not correspond to

evaluations of F anymore, but to evaluations of descendants of F, which are functions acting on a number of variables between n and $n - \ell$. The complexity of these attacks depends on the properties of the descendants rather than the ones of F. In the security analysis of [36], the descendant with the worst parameter was considered for each Boolean criterion, giving a lower bound on the complexity of the corresponding attack. By randomizing the choice of the descendant, the whitening enables the security of IFPs to be based on average properties rather than worst-case ones (as the probability of getting a function with the worst parameters is not equal to 1).

Finally, note that increasing the register size makes the construction very similar to Goldreich's PRG [25]. For more details on this PRG, we refer to the initial article of Goldreich and to the survey of Applebaum [5]. In the following we give the necessary explanations to understand the connection between this PRG and IFPs. Goldreich's PRG is an asymptotic construction with interesting conjectured security [4, 5, 6], and many implications such as secure computation with constant computational overhead [29], or indistinguishability obfuscation [32,33]. We can define this PRG in the following way: let n and m be two integers, let $(S_1, \ldots, S_m)$ be a list of m subsets of $[n]$ of size d, and let $P \in \mathcal{B}_d$ (often called predicate), we call Goldreich's PRG the functions $G : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$ such that for $x \in \mathbb{F}_2^n$, $G(x) = P(S_1(x)), P(S_2(x)), \ldots, P(S_m(x))$. The integer d is called the locality of the PRG and many works have focused on polynomial-stretch local PRG. Local means that d is a constant, and polynomial-stretch means that $m = n^s$ where s is the called the stretch: these PRG extend a short random seed into a polynomially longer pseudorandom string. These local PRG are conjectured secure based on some properties of the subsets and on the function P. Considering the (n, m, d)-hypergraph given by the subsets $(S_1, \ldots, S_m)$, the PRG cannot be secure if the hypergraph is not sufficiently expending (we refer to [5] for the notions and references). In practice, an overwhelming portion of (n, m, d)-hypergraphs are sufficiently expanding, making the choice of a random (n, m, d)-hypergraph an usual and adequate strategy. For the function P, the PRG cannot be secure if P is not resilient enough [39] or if its algebraic degree, or more generally its algebraic immunity, is insufficient [6], both quantity being related to s. For these constructions, the security is considered asymptotically, relatively to classes of polynomial adversaries as linear distinguishers [39] or the Lasserre/Parrilo semidefinite programming hierarchy. Regarding concrete parameters, very few is known up to now, we are only aware of the recent work [17], which concretely studies the security of an instance of a super-linear (but less than quadratic) stretch.

3.3 Impact on Homomorphic Evaluation.

The modifications from FPs to IFPs are almost free. The size of the key register does not modify the function F so the homomorphic error-growth given by the evaluation of F is independent of N. The whitening is given by the output of the PRNG, so considered as public, therefore each bit of the whitening is encrypted as a zero-noise homomorphic ciphertext. Adding homomorphically these zero-noise ciphertexts to the input of F does not increase the error-growth, giving a final noise identical to the one obtained with a FP instantiated with the same function. Only the time of the evaluation is modified, but the search in a longer list and the addition of zero-noise ciphertexts has a minor impact compared to the evaluation of the filtering function.

3.4 Key-size Consideration.

A general idea behind FPs and Improved FPs is to have the main part of the encryption process which would have no cost when homomorphically evaluated. This specificity leads to consider longer keys than the traditional λ-bits key for a bit-security of λ. We argue that in the SE-FHE context this specificity has a very low impact. Indeed, even bounding the total key-size to 2^{14} it is still way smaller that the size of only one homomorphic ciphertext. Then, the encryption of each bit depending only on a subpart of fixed length of the key, the total length of the key has no impact for the majority of the hybrid FHE framework. Since the user can store a key of this size, and the server can store this amount of homomorphic ciphertexts, the key size is not a bottleneck in the considered framework. Note that for the schemes with key size of λ bits, more computations are needed for the encryption or decryption, having an important impact on the size of the homomorphic ciphertexts required, impacting the majority of the hybrid FHE framework, and mostly the application part.

4 Security Analysis of the Improved Filter Permutators.

Due to the similarity of (improved) filter permutators to the filter register model, we investigate the attacks known to apply on this model. We consider that no additional weakness arises from the PRNG which is chosen to be forward secure to avoid malleability. The subsets and the whitenings are chosen without any bias, and Knuth-shuffle is used to choose the permutations. As a consequence, on this pseudorandom system non adversarially chosen, the attacks applying target the filtering function and they are adaptations from the one applying on filtered registers. The first part of the security analysis is similar to the one in [36], the same kind of attacks are explored but the complexity is computed differently, using new algorithms enabling to consider any functions. We consider the attacks in the single-key setting, in the known ciphertext model, focusing particularly on key-recovery attacks.

4.1 Algebraic-like Attacks.

We qualify as algebraic-like attacks the kind of attacks consisting in manipulating the system of equations given by the key-stream to build a system of smaller degree, easier to solve. Algebraic attacks [16], fast algebraic attacks [14], or approaches using Grobner bases (such as [21]) are examples of this type. To determine the security of IFP relatively to this class of attacks we study more particularly the complexity of algebraic and fast algebraic attacks, as their complexity can be estimated from Boolean criteria.

The main idea of algebraic attacks as defined in [16] (in a context of filtered LFSR) is to build an over-defined system of equations with the initial state of the LFSR as unknown, and to solve this system with Gaussian elimination. The principle is to find a nonzero function g such that both g and $h = gF$ have low algebraic degree, allowing the attacker to get various equations of small degree d. Then, the degree-d algebraic system is solved, by linearization if it is possible, using Grobner basis method or SAT solvers otherwise; linearization is the only method for which evaluating the complexity is easy. The degree of g is at least $\mathsf{AI}(F)$, and g is chosen to be a non null annihilator of F or $F + 1$ of minimal degree. Then the adversary is able to obtain $\mathcal{D}\mathsf{AN}(F)$ (respectively

$\mathcal{D}\mathsf{AN}(F+1)$) equations with monomials of degree $\mathsf{AI}(F)$ in the key bits variables, for each equation. After linearization, the adversary obtains a system of equations in $D = \sum_{i=0}^{\mathsf{AI}(F)} \binom{N}{i}$ variables, where N is the number of original indeterminates. Therefore, the time complexity of the algebraic attack is $\mathcal{O}(D^\omega) \approx \mathcal{O}(N^{\omega \mathsf{AI}(F)})$, where ω is the exponent in the complexity of Gaussian elimination (we assume $\omega = \log(7)$ for all our security estimations [3]). The data complexity is $\mathcal{O}(D/\mathcal{D}\mathsf{AN}(F))$.

Fast algebraic attacks [14] are a variation of the previous attacks. Still considering the relation $gF = h$, their goal is to find and use functions g of low algebraic degree e, possibly smaller than $\mathsf{AI}(f)$, and h of low but possibly larger degree d. Then, the attacker lowers the degree of the resulting equations by an off-line elimination of the monomials of degrees larger than e (several equations being needed to obtain each one with degree at most e). Following [7], this attack can be decomposed into four steps. The search for the polynomials g and h generating a system of $D + E$ equations in $D + E$ unknowns, where $D = \sum_{i=0}^{d} \{N \text{ choose } i\}$, and $E = \sum_{i=0}^{e} \{N \text{ choose } i\}$. The search for linear relations which allow the suppression of the monomials of degree more than e, with a time complexity in $\mathcal{O}(D \log^2(D))$. The elimination of monomials of degree larger than e using the Berlekamp-Massey algorithm, corresponding to a time complexity in $\mathcal{O}(ED \log(D))$. Finally, the resolution of the system, in $\mathcal{O}(E^\omega)$. This attack is very efficient on filtered LFSR ciphers as the search of linear relations between equations is simple. For IFPs, the first step could be trivial for our choice of F. Then, as the subset of variables and the permutation chosen at each clock cycle are given by the PRNG, there is no trivial linear relation between one equation and the next ones. It is always possible to simplify some equations using the system, for example forcing collisions on the monomials of higher degree, so other techniques of eliminations could apply. We stress that the time complexity of these techniques would be higher than the one of Berlekamp-Massey. Thus we consider the (time) complexity of the fast algebraic attack, $O(D \log^2(D) + ED \log(D) + E^\omega) \approx \mathcal{O}(N^{\mathsf{FAI}})$, as an upper bound on the time complexity of any attack of the algebraic kind on IFPs, and a data complexity of D.

4.2 Correlation-like Attacks.

We qualify as correlation-like attacks the kind of attacks that use the bias of the filtering function relatively to uniform, or to a low degree function. Correlation attacks, Learning Parity with Noise (LPN) solvers, correlations based on the XL algorithm [15] are examples of this kind. To determine the security of IFP relatively to this class of attacks, we study more particularly the complexity of correlation attacks, and show how it complexity can be estimated using Boolean criteria.

The principle of correlation attacks is to distinguish the output of IFPs from random. For example if the filtering function is unbalanced an attack can consist in averaging the key-stream and observing a bias towards $1/2$. If the function is balanced, this strategy does not apply, but instead of averaging on all the key-stream, the attack can target one sub-part only, depending on a small portion of the variables. As the goal of these attacks

[3] for a sparse system of equations we could use $\omega = 2$, but, note that even if the filtering function has a sparse ANF, it does not imply that this property holds on its annihilators. Then, the systems targeted by the (fast) algebraic attacks have a lower degree but are denser, justifying this more common choice for ω.

is to distinguish the key-stream from random, then we assume that key-recovery attacks have at least the same complexity. Two points influence the effectiveness of this attack: the possibility to get equations relatively to an unbalanced function, and the bias.

Two criteria enable us to study the functions relatively to these points: the resiliency and the nonlinearity. The resiliency of a function gives the number of variables that have to be fixed to make it unbalanced, and can be used for the first point. Then, the nonlinearity gives the distance with the closest affine function, determining the bias to $1/2$. Note that to detect the bias to $1/2$ the data complexity would be $\mathcal{O}(\delta^{-2})$, with $\delta = 1/2 - \mathsf{NL}(F)/2^n$. For LPN solvers, correlation based on XL, or other attacks of this kind, a similar bias has to be observed. The smaller is δ, the more distant is the algebraic system from a linear one, which decreases the efficiency of these attacks. When combinations of vectors are required to observe a bias, the higher is the resiliency, the higher is the attack complexity. We adopt a conservative approach to thwart this variety of attacks: we assume that guaranteeing both $\delta^{-2} \geq 2^\lambda$ and a resiliency of $\lambda - 1$ avoids any attack of this kind with time or data complexity of less that 2^λ operations.

Note that in the context of Goldreich's PRG only the resiliency is studied. The underlying principle is, as the output is bounded (polynomial) and as the subsets are well distributed, the probability of repetitively finding subsets of the key-stream bits whose sum gives an unbalanced function is low, with enough resilience. In this context the nonlinearity is not studied, as any bias is considered as giving a polynomial attack.

4.3 Guess-and-determine Strategies.

As shown in [20] guess-and-determine attacks apply on FPs. Thus, we consider this class of attacks relatively to IFPs. The principle of the guess-and-determine attack consists in guessing ℓ key bits in order to target simpler functions, obtaining a system of equations easier to solve or with a distribution easier to distinguish. In our context it can be less costly for an attacker to consider the 2^ℓ possible systems given by fixing the value of ℓ variables than attacking the initial system of equation given by the key-stream. Hence, both kinds of attacks presented before can be generalized with guess-and-determine. We explain the principle relatively to the algebraic attack: the attacker selects ℓ variables and gives a value of its guess, it simplifies the algebraic system. Then, the attacker considers all equations such that the descendant function has algebraic immunity at most k, and generates the corresponding degree k algebraic system. Once linearized, the attacker solves the system, if it is not consistent, then another guess is tried. As one of the 2^ℓ values of the guess is the correct one, the attack will succeed. Similarly for the other attacks, once the value of the guess is fixed, the attack is mounted on the new system relatively to a specific value of a parameter (the value of e and d for the fast algebraic attack, the value of δ, or the value of the resiliency).

A bound on the complexity of these attacks can be derived from the complexity of the attack without guess-and-determine. For the time complexity, it corresponds to multiplying by 2^ℓ the complexity of the attack using the parameter of value k on a system with $N - \ell$ variables. For the data complexity, the probability of getting a function with parameter k is important, the whole complexity can then be bounded by the inverse of this probability multiplied by the complexity of the attack using the parameter of value k on a system with $N - \ell$ variables. To determine this probability, it requires to determine the parameters relatively to the Boolean criteria of all descendant

functions of F up to $\ell \leq \lambda$ variables. Some descendants may have extreme parameters (called recurrent criteria in [36]), but very low probability of appearing. Then for attacks with guess-and-determine, it is important to investigate both time and data complexities.

Note that the particular guess-and-determine attacks investigated in [20] on FLIP uses two properties. First, the fact that the filtering functions has a very sparse ANF so the descendant obtained by fixing zeros is the one with worse parameters, and then, the crucial fact that for filter permutators the guesses made on the key bits are directly in input of F. The whitening changes the latter property, then targeting a weak function requires to have both the guesses and the whitenings coinciding. It leads to an higher data complexity, as shown by the algorithms estimating the complexity at the end of this section.

4.4 Other Attacks.

Besides the previous attacks that will be taken into account quantitatively when selecting concrete instances, we also investigated other cryptanalyses, so we develop some explanations on those which are known to apply on filter permutators [36].

First, weak key attacks can be considered: if the Hamming weight of the key is extreme the input of F is far away from the uniform distribution. The probability of this weight to be extreme is very low due to the register extension, and as explained before the whitening avoids simple attacks using the unusual distribution of F's inputs. Restricting our instances to keys of Hamming weight $N/2$ handles these attacks. Second, higher-order correlation attacks [15] consist in approximating the filtering function by a function of degree $d > 1$ and to solve the approximated algebraic system of degree d with a Grobner basis algorithm such as F4 [21]. The attack could be efficient if the function was very close to a degree d function (which corresponds to a small nonlinearity of order d), and if d was low enough as one part of the attack consists in solving a degree d system. This attack can easily be combined with guess-and-determine techniques, but up to now for the filtering functions we tried, the complexity of this attack is always superior to the one considered for fast algebraic attacks or for correlation-like attacks. Eventually, restricted input attacks [12] using the behavior of F on a restricted part of its input are handled by the register size and the whitening. Since the input of F is not restricted to a subset of $\mathbb{F}_2^n$, but to the whole set, it seems unrealistic to adapt this attacks in this context. It would require to combine equations to obtain a set of equations corresponding with high probability to a known small subset of $\mathbb{F}_2^n$. Moreover the function should also have some flaws relatively to this particular subset, which we leave as a scope for further investigations.

4.5 Estimating the Attacks Complexity.

Based on the previous parts of this section, relatively to a Boolean function F and the register size N, we can estimate the security of IFPs by computing the parameters of each descendant up to λ variables. We describe the principle of the algorithm used to determine the complexity of an attack relatively to a parameter. To illustrate it, we consider (a simplified version for) the attack based on the algebraic immunity.

Algorithm's Principle. Determining the security from any filtering function F:
1. From F and λ, the profile of the function relatively to algebraic immunity is computed. The profile corresponds to the probability of getting a descendant of F with algebraic immunity less than or equal to k ($0 \leq k \leq \mathsf{AI}(F)$) by fixing ℓ bits of F inputs. The probability is taken over all choices of ℓ over n variables ($0 \leq \ell \leq \lambda$) and over the 2^ℓ possible values taken by these variables. To compute the profile, the probability of getting each descendant is computed iteratively, from step 0 to λ. Step 0 corresponds to the function F with probability 1, the profile for 0 guess gives a probability of 0 for $k < \mathsf{AI}(F)$ and 1 for $k \geq \mathsf{AI}(F)$. Then, from step ℓ to step $\ell + 1$, for each descendant of step ℓ and its probability, all descendants obtained by fixing one of its variables (to 0 and to 1) are computed, together with their probability. It gives then all descendants of step $\ell + 1$, the algebraic immunity of each one is computed, and the profile for ℓ guesses at value k is the sum of the probabilities of all these descendants with algebraic immunity less than or equal to k.
2. From the profile and N, for each L with $0 \leq L \leq \lambda$, and for each possible value k of the algebraic immunity ($0 \leq k \leq \mathsf{AI}(F)$), we compute the time and data complexity of the attack targeting functions with AI less than or equal to k. The time complexity is then 2^L multiplied by the time complexity of an algebraic attack with AI equal to k on a system in $N - L$ variables. The data complexity depends on the probability of obtaining an equation with such a parameter of AI. This probability depends on the profile and on N. It corresponds to $P = \sum_{\ell=0}^{L} P_{L=\ell} \cdot P_{(\mathsf{AI} \leq k) \,|\, \ell}$, where $P_{L=\ell} = \binom{L}{\ell}\binom{N-L}{n-\ell}\binom{N}{n}^{-1}$ is the probability that ℓ over the L guesses of the adversary are in the n input's variables of F. The quantity $P_{(\mathsf{AI} \leq k) \,|\, \ell}$ is the probability that the function has algebraic immunity less than or equal to k conditioned on the number of variables fixed in F to get this function. This probability is what the profile gives. The data complexity is finally P^{-1} multiplied by the data complexity of an algebraic attack with algebraic immunity equal to k on a system in $N - L$ variables.
3. For each pair (L, k), we determine the maximum between the time and data complexity, the minimum over all pairs gives the final complexity of the attack.

Potential Modifications. The advantage of this methodology is to apply on any filtering function F, and any register size N, giving a general framework to determine the security of IFPs instances. This general algorithm being exhaustive, it has a high time and storage complexity. Indeed, note that the number of descendants of a function is exponential. The algorithm can be modified in order to be more efficiently evaluated, but sometimes at the cost of underestimating the cost of the attacks.

A first modification, which does not underestimates the cost of the attacks, consists in finding the descendants which are equivalent. That is, the ones which have exactly the same parameters for each criterion and that give the same descendants with identical probabilities. When such equivalent descendants are found, which can be handled through the representation of the function, the number of descendant at step ℓ can be less than the initial bound of $2^\ell\binom{n}{\ell}$. A second modification, underestimating the cost of the attacks, consists in replacing each value of the parameter (which can take in some cases numerous values) by the nearest one among those which are more favorable to the attacker in a shorter list, and summing the probabilities corresponding to each such approximation. A third modification, also underestimating the cost of the attacks,

can be achieved by not considering all descendants but only descendants which have worse parameters. It is possible when, for each number of guesses considered, for each criterion, the profile of a function is worse than the profile of another one. Then the probability of the function with better profiles can be added to the probability of the function with worse profiles. In other words, a (stronger) function can be neglected and its probability added to another one, if the probability of its descendants to reach a particular weak value of parameter is always inferior than the corresponding probability for the descendants of the other (weaker) function.

5 Instantiating the Improved Filter Permutators with DSM Functions: $\mathsf{FiLIP}_{\mathsf{DSM}}$ Instances.

We now instantiate the IFP paradigm with filtering functions being direct sums of monomials, and denote these instances as $\mathsf{FiLIP}_{\mathsf{DSM}}$. This choice is motivated by the functions considered in [36] under the name of FLIP functions, which are a sub-family of DSM functions. DSM functions are very structured functions, are easy to represent through their direct sum vector, and we can determine all their parameters relatively to Boolean criteria. Recall that to estimate the most correctly the security given by a filtering function, it is necessary to determine the parameters of all its descendants (up to λ variables). As DSM are bit-fixing stable (see remark 1), knowing the standard properties of all the family enables to determine the bit-fixing properties of any DSM filtering function, giving a very accurate estimation of the security against guess-and-determine attacks. Finally, it is a good choice in terms of homomorphic error-growth due to their low multiplicative depth, which is considered as the main parameter influencing second generation FHE schemes such as BGV [9]. Evaluating DSM functions consists in summing products of binary ciphertexts only, which corresponds to a very low error-growth, both in second, and in third generations. More precisely, the evaluation of a DSM function with a 3G scheme produces only a quasi-linear (in n) error-growth ([35], Lemma 22).

We instantiate the forward secure PRNG following Bellare and Yee [8] construction, using the AES as underlying block cipher. The PRNG is set with two constants C_0 and C_1, For each K_i the first block $AES(K_i, C_0)$ gives the key K_{i+1} of the next iteration and the second block $AES(K_i, C_1)$ gives 128 bits being the i-th part of the PRG's output. For each key-stream bit the PRNG outputs $\lceil \log \binom{N}{n} \rceil$ bits used to select the subset considering the variables in lexicographic order. Then, the permutation over n bits is instantiated with the Knuth shuffle [30] with the following bits output by the PRNG. Finally, n last bits are used to generate the whitening. If the number of ciphertext bits $m \leq 2^\lambda$ requires more pseudorandom bits that the secure limitation of the PRNG, another instance is used with other constants. The number of possible instances for the PRNG makes that for the parameters we consider the limitation comes from m.

5.1 Simplifying the Attack Complexities Algorithms for DSM Functions.

In Section 4 we give the general framework to compute the complexity of the attacks. Even if DSM functions form a bit-fixing stable family, computing exactly the parameters of all descendants of a non trivial DSM in more than 300 variables is out of reach. Then, we use general properties proven on these functions in [35] to modify the

algorithms as explained in Section 4.5. We describe in the following these modifications which greatly improve the complexity and finally enable to find concrete instances.

First note that the criteria of resiliency, nonlinearity, algebraic immunity and fast algebraic immunity can exactly be determined using the DSV notation (for the $\mathcal{D}\mathsf{AN}$ the constant term of the function matters). Therefore, two functions with the same DSV are considered as equivalent, and the number of descendants to consider decreases using this property, it corresponds to the first modification mentioned in Section 4.5. Then, the number of descendants with different DSV is still very important, and the number of different parameters also. consequently, we use the second modification relatively to the nonlinearity, the $\mathcal{D}\mathsf{AN}$ and the fast algebraic immunity. For DSM functions, the exact value of the bias δ varies a lot, hence we consider only the values of $-\lfloor(\log(\delta)\rfloor$. For the $\mathcal{D}\mathsf{AN}$ we use an upper bound (compatible with the DSV notation), considering the maximum over all DSM of degree at most k. For fast algebraic attack we consider only 1 and 2 as possible values for e and the reached algebraic immunity as possible value for d. Finally, we decide to not consider all descendants, and attribute the probability of the ones with good parameters to others (third modification). It is the modification affecting most the complexity of the algorithm. It is realized through proving relations between the parameters of DSM functions and an order on their DSV. In the case of DSM, the descendants obtained by fixing zeros are always with worse parameter.

First Modification, Parameters Given by the Direct Sum Vector. As far as we know DSM functions constitute the first bit-fixing stable family for which all the parameters (mentioned in Section 4) are determined. Note that for a DSM function f with $\mathbf{m}_f = [m_1, m_2, \ldots, m_k]$ there are at most $M = \prod_{i=1}^{k}(m_i + 1)$ different DSV in the descendants obtained by fixing zeros (as fixing a variable to 0 decreases one of the m_i by 1). To compute the profiles of a DSM, we use the following representation, as only the descendants obtained by fixing zeros are considered, we store a vector of length M and each index represent one descendant. The number of descendants at each step being the most expensive part of the algorithm in term of storage and time, the algorithm is better suited for function with relatively small M. It justifies why we will focus on instances with sparse DSV.

Lemma 1 (Direct Sum of Monomials and Boolean Criteria, [35] Lemmata 2, 3 and Theorem 1). *Let $f \in \mathbb{F}_2^n$ be a Boolean function obtained by direct sums of monomials with associated direct sum vector $= [m_1, \ldots, m_k]$, its parameters are:*

- *Resiliency:* $\mathsf{res}(f) = m_1 - 1$.
- *Nonlinearity:* $\mathsf{NL}(f) = 2^{n-1} - \frac{1}{2}\left(2^{(n-\sum_{i=2}^{k} i m_i)} \prod_{i=2}^{k} \left(2^i - 2\right)^{m_i}\right)$.
- *Algebraic immunity:* $\mathsf{AI}(f) = \min_{0 \le d \le k}\left(d + \sum_{i=d+1}^{k} m_i\right)$.

Second Modification, Bounding FAI and $\mathcal{D}\mathsf{AN}$. We recall the lower bound on the FAI and the upper bound of the $\mathcal{D}\mathsf{AN}$ of a DSM in the following proposition. The bound on the FAI is tight for most functions, whereas the situation is opposite for the bound on

the $\mathcal{D}\mathsf{AN}$. It is very common that all descendant of a DSM with a different DSV have a different value of $\mathcal{D}\mathsf{AN}$. Hence, the bound allows us to reduce from $\prod_{i=1}^{k}(m_i+1)$ possible values to $2k$ only. This modification simplifies greatly the estimation of the algebraic attack complexity, as it requires to jointly use AI and $\mathcal{D}\mathsf{AN}$ profiles.

Proposition 1 (Bounds on FAI and $\mathcal{D}\mathsf{AN}$, [35] Propositions 4 and 5). *For any DSM function f of degree $k>0$:*

$$\mathsf{FAI}(f) \geq \mathsf{bFAI}(f) = \begin{cases} \mathsf{AI}(f)+2 & \text{if } \mathsf{AI}(f) = \deg(f), \mathsf{AI}(f) > 1, \text{and } m_k > 1, \\ \mathsf{AI}(f)+1 & \text{otherwise.} \end{cases}$$

$$\mathcal{D}\mathsf{AN}(f) \leq \mathsf{b}\mathcal{D}\mathsf{AN}(f) = k^k+1 \text{ if } m_1 = 0, \text{ and } k^{k-1}+1 \text{ if } m_1 > 0.$$

Third Modification, Determining DSM Descendant with Worse Properties. In the previous paragraphs we consider the number of descendants obtained by fixing zeros and not the total number of descendants, the reason is that a relation between the descendant greatly decreases the algorithms complexity. We use this relation between the 2^ℓ descendants to bound the parameters relatively to the standard criteria of all descendant functions on a same subset by the parameters of only one of these descendant. The purpose of these results is to be able to upper bound the number of descendants of a function which parameter is equal to a targeted value. More specifically, for all $b \in \mathbb{F}_2^\ell$, the parameters of all descendant functions relatively to the same subset can be bound using the direct sum vectors of the all-0 or all-1 descendant:

Proposition 2 (DSM Descendants Properties, [35] Proposition 6). *Let $\ell \in \mathbb{N}^*$, 0^ℓ and 1^ℓ denote the all-0 and all-1 vectors of $\mathbb{F}_2^\ell$ .Let f be a DSM function in n variables, for any $\ell \,|\, 1 \leq \ell < n$, and any subset $I \subseteq [n]$ such that $|I| = \ell$, and for any $b \in \mathbb{F}_2^\ell$:*

$$\mathsf{res}(f_{I,b}) \geq \mathsf{res}(f_{I,0^\ell}), \quad \mathsf{NL}(f_{I,b}) \geq \mathsf{NL}(f_{I,0^\ell}), \quad \mathsf{AI}(f_{I,b}) \geq \mathsf{AI}(f_{I,0^\ell}),$$
$$\mathsf{bFAI}(f_{I,b}) \geq \mathsf{bFAI}(f_{I,0^\ell}), \quad \mathsf{b}\mathcal{D}\mathsf{AN}(f_{I,b}) \leq \mathsf{b}\mathcal{D}\mathsf{AN}(f_{I,1^\ell}).$$

5.2 Concrete Instances with DSM Functions.

Based on the security estimations we propose the following instances of $\mathsf{FiLIP}_{\mathsf{DSM}}$ in Table 1, $\mathbf{m}_F$ is the DSV notation of F, n is the number of variables of F, N is the size of the key register, d is the multiplicative depth of the function, and λ is the conjectured security parameter.

$\mathbf{m}_F$	n	N	d	λ	Name
$[89,67,47,37]$	512	16384	2	80	FiLIP-512
$[80,40,15,15,15,15]$	430	1792	3	80	FiLIP-430
$[80,40,0,20,0,0,0,10]$	320	1800	3	80	FiLIP-320
$[128,64,0,80,0,0,0,80]$	1216	16384	3	128	FiLIP-1216
$[128,64,0,0,0,0,0,0,0,0,0,0,0,0,0,64]$	1280	4096	4	128	FiLIP-1280

Table 1. $\mathsf{FiLIP}_{\mathsf{DSM}}$ Instances.

6 Performance Evaluation.

Ultimately, the goal of SE-FHE applications is to obtain the result of the homomorphic computations with the best latency and throughput. However, such performance metrics can only be evaluated if the functions to be evaluated by the Cloud are known in advance. In previous evaluations of symmetric ciphers for FHE evaluation, this issue was (partially) circumvented by studying the latency and throughput of homomorphic ciphertexts that will just enable decryption or a fixed number of levels of multiplications. This allows getting lower bounds on the timings necessary to evaluate any function, and the performances are reasonably accurate for simple functions with the given multiplicative depth. Yet, one important drawback of this approach remains that optimizing latency and throughput requires to fix parameters such as the size of the ciphertexts and the quantity of noise (which set the security of the FHE scheme). More precisely, in HE, it is the quantity of noise that determines the size of the ciphertexts required to correctly handle the operations. This size is in turn the main factor determining the latency and throughput of the homomorphic operations. Therefore, optimizing throughput and latency is ideal for one specific function, but it looses its accuracy when the application deviates from this particular function. We next propose an alternative comparison methodology, based on the homomorphic noise, that aims to be more independent of the applications.

6.1 Methodology.

Considering the performances of SE-FHE relatively to the homomorphic noise is based on two simple principles. The smaller is the noise, the wider is the class of functions still evaluable on these ciphertexts. The smaller is the noise, the smaller are the homomorphic ciphertexts, the faster are the evaluations. It means that the homomorphic noise dictates the ciphertext parameters, and eventually the latency and throughput of the final application. Consequently, an appealing performance evaluation could consist in determining exactly the error-growth (in average or with overwhelming probability) given by an SE scheme for a specific FHE scheme. As there is no simple parameter (such as the multiplicative depth) which encompasses totally the error-growth, we use a simpler methodology consisting in measuring the noise just after evaluating the symmetric decryption or after some additive levels of multiplications.

In contrast with the aforementioned latency/throughput oriented methodology, which leads to fix the homomorphic parameters to optimize the timings for a given target function, a noise-oriented methodology can ensure that the ciphertext parameters are the same for all SE schemes to be compared. This has two advantages. First, all homomorphic ciphertexts obtained have the same security, that we fix to λ, the security level of the SE scheme. Second, once the symmetric decryption is performed, the evaluation time of any function will be independent of the SE scheme used for the transciphering. Such a scheme is then only limited by the ciphertext noise, which determines the quantity of operations that can be performed until decryption becomes impossible. Consequently, with this methodology, the smaller is the measured noise, the better suited is the SE scheme. We believe this approach provides interesting insights for the comparison of SE schemes in an application-dependent manner.

Additionally, and for completeness, we give some indications on the time performances, using the strategy of previous works. To do so, for each SE scheme we select homomorphic parameters that are sufficient to evaluate the decryption circuit, but no more. It gives an idea on the minimal size of homomorphic ciphertext and minimal evaluation time required for each SE scheme relatively to the library used. The result corresponds to a minimum as for any application, bigger ciphertexts are necessary to make the evaluations of the computation part.

6.2 Performances and Comparisons.

We chose to compare the following symmetric schemes: LowMC [2], FLIP [36], Rasta and Agrasta [19] and FiLIP$_{\mathsf{DSM}}$, all designed for the SE-FHE framework. We did not consider Kreyvium [10] as its implementation is very different (based on previous studies the related numbers would be slightly better than the one of LowMC due to a multiplicative depth of 12 and 13). All implementations were made with the HElib library [28]. The LowMC implementations were taken from the publicly available code (`https://bitbucket.org/malb/lowmc-helib`). The one of Rasta were built from this implementation and the publicly available one (`https://github.com/iaikkrypto/rasta`). We use the same code for computing the "Four Russians" method, used for multiplying binary matrices. The FLIP and FiLIP$_{\mathsf{DSM}}$ implementations were made ad hoc. These implementations were evaluated on laptop computer with processor Intel(R) Core(TM) i5-4210M CPU at 2.60GHz.

Accordingly to the previously described methodology, we chose parameters in HElib enabling to evaluate the decryption of all these schemes. These parameters are dictated by LowMC (due to its higher multiplicative depth), so we choose the minimal parameters such that LowMC ciphertexts can be decrypted while keeping an FHE security of at least the security of the symmetric ciphers. The noise level after evaluation is estimated thanks to HElib function log_of_ratio() which returns the difference $\log(\sigma) - \log(q)$ where σ^2 is the noise variance (derived from bounds on the error-growth of addition, product, automorphism, and switchings) of the error part of the ciphertext, and q is the modulus. In order to have a glimpse of what this noise level represents, we also computed 1 (respectively 2) level(s) of multiplications between ciphertexts (after the homomorphic evaluation of the symmetric schemes). The results for 80-bit security and 128-bit security are given in Table 2 and Table 3[4]. Symbol d denotes the multiplicative depth of the decryption circuit of the SE scheme, N is the key size, symbol b denotes the number of produced bits. The latency refers to the time required to have the first ciphertext after evaluation, the noise columns refer to the output of the log_of_ratio() function, with respectively 0, 1 and 2 levels of multiplications (after evaluation of the SE decryption function).

From these results we can conclude that LowMC ciphertexts have the biggest error-growth. For the 80-bit security instances the noise after evaluating FLIP, Rasta or Agrasta is similar whereas the instances of FiLIP$_{\mathsf{DSM}}$ enable 1 or 2 additional levels of multiplications. For 128 bits of security, FiLIP-1280 ciphertexts are slightly less noisy than Agrasta and FLIP ciphertexts, whereas FiLIP-1216 offers an additional level of multiplications. In terms of evaluation time, the parameters are more suited

[4] These security levels are the one given by HElib, more accurate estimations are given in [1].

Cipher	d	N	b	Latency (s)	noise	noise $\times$	noise $\times^2$
LowMCv2 (12, 31, 128)	12	80	128	329.38	-2.966	n/a	n/a
LowMCv2 (12, 49, 256)	12	80	256	699.10	-2.495	n/a	n/a
Agrasta (81, 4)	4	81	81	67.48	-155.722	-139.423	-119.459
Rasta (327, 4)	4	327	327	290.99	-154.502	-139.423	-119.459
Rasta (327, 5)	5	327	327	366.30	-135.727	-119.459	-100.641
FLIP-530	4	530	1	42.06	-157.201	-139.423	-119.459
FiLIP-512	2	16384	1	33.74	-194.342	-177.739	-158.241
FiLIP-430	3	1792	1	31.25	-176.039	-158.241	-139.423
FiLIP-320	3	1800	1	21.41	-176.588	-158.241	-139.423

Table 2. Noise comparison for 80-bit security. HElib parameters: LWE dimension 15709, HElib Depth L 14, B = 28 (Bit per level parameter that influence BGV security), BGV security 84.3, Nslots 682, log_of_ratio() of fresh ciphertext -237.259.

Cipher	d	N	b	Latency (s)	noise	noise $\times$	noise $\times^2$
LowMCv2(14, 63, 256)	14	128	256	1629.03	-3.418	n/a	n/a
Agrasta (129, 4)	4	128	129	207.68	-207.478	-190.086	-169.011
Rasta (525, 5)	5	525	525	1264.30	-185.885	-169.011	-148.313
Rasta (351, 6)	6	351	351	967.62	-164.945	-148.313	-129.716
FLIP-1394	4	1394	1	272.31	-207.831	-190.086	-169.011
FiLIP-1216	3	16384	1	251.28	-227.93	-210.437	-190.086
FiLIP-1280	4	4096	1	325.04	-208.112	-190.086	-169.011

Table 3. Noise comparison for 128-bit security. HElib parameters: LWE dimension of the underlying lattice = 24929, HElib Depth L = 16, B = 30, BGV security = 132.1, Nslots =512, log_of_ratio() of fresh ciphertext = -293.929.

for LowMC, but relatively to this size of ciphertexts we can conclude that Agrasta evaluations produce more ciphertexts per second. The instances of $\mathsf{FiLIP}_{\mathsf{DSM}}$ produce the ciphertexts one by one, and have then a throughput around 50 times slower for 80-bit instances and 200 for 128-bit instances. These results confirm the excellent behavior of $\mathsf{FiLIP}_{\mathsf{DSM}}$ in terms of noise, enabling 1 or 2 supplementary levels of multiplication (at the cost of a moderate decrease of the time performances detailed next).

Note that the gain in depth of $\mathsf{FiLIP}_{\mathsf{DSM}}$ relatively to Agrasta or Rasta is obtained at the price of larger key sizes. When choosing which scheme to use in the hybrid homomorphic framework, a trade-off can be considered between these schemes, depending on the number of levels of multiplications required (computation phase) and constraints on the key-size (initialization phase). The more computations over the data will be considered, the more important will be the influence of the error-growth, making negligible the impact of the key-size.

We also note that in [19], instances with a smaller multiplicative depth are considered, but the authors recommend a depth at least 4 for security reason. These instances always involve way bigger keys than $\mathsf{FiLIP}_{\mathsf{DSM}}$ instances with the same

multiplicative depth, and due to the high number of XORs in these instances, the error-growth is higher. Rasta ciphers were not optimized for the metric we consider, instances designed for the error-growth could lead to better performances. We argue that minor modifications would benefit to evaluation over HElib, but by design the noise remain larger than the one from IFPs. For example, the high number of additions occurring at different levels between multiplications prohibits Rasta design to be used in a SE-FHE framework using 3G FHE, whereas IFPs are performing well for all known FHE.

We also study the performance results in time for the different SE ciphers considered. For this purpose, we chose the HElib parameters such that the ciphers can just be decrypted (by setting the appropriate L value), while keeping a similar security level for the HE scheme (by modifying with trial and errors the other parameters). These numbers have to be taken as a global behavior of the achievable performances of the ciphers. Optimizations can still be made in the code itself but also in the choice of the FHE parameters. We report the results in Table 4, B is the bit per level parameter, m is the LWE dimension, L is the HElib depth, λ' is the BGV security estimated by HElib, ns the number of slots. The latency refers to the time required to have the first ciphertext after evaluation, the noise columns refers to the output of the log_of_ratio() function after evaluation of the decryption. These results show that, adapting the FHE parameters to the decryption of the SE scheme only, the throughput can be sensibly increased. Note that, for some lines the ciphertexts are still usable for more evaluations, it comes from the fact that HElib rejects smaller values of L, whereas the multiplicative depth of the scheme is inferior.

Cipher	B	m	L	λ'	ns	Latency (s)	noise
LowMCv2(12, 31, 128)	28	15709	14	84.3	682	329.38	-2.966
LowMCv2(12, 49, 256)	28	15709	14	84.3	682	699.10	-2.495
Agrasta (81, 4)	26	5461	5	82.9	378	12.97	-2.03
Rasta (327, 4)	26	8435	5	84.6	240	76.33	-1.903
Rasta (327, 5)	25	7781	7	85.1	150	90.78	-14.42
FLIP-530	21	4859	5	85.3	168	6.48	-1.23
FiLIP-512	21	4859	5	85.3	168	7.05	-29.09
FiLIP-430	21	4859	5	85.3	168	6.01	-15.457
FiLIP-320	21	4859	5	85.3	168	5.04	-16.02
LowMCv2(14, 63, 256)	30	24929	16	132.1	512	1629.3	-3.418
Agrasta (129, 4)	27	7781	5	134.7	150	20.26	-3.03
Rasta (525, 5)	27	10261	7	128.9	330	277.24	-20.441
Rasta (351, 6)	27	10261	8	128.9	330	195.40	-1.92
FLIP-1394	28	8191	6	146.8	630	26.53	-5.11
FiLIP-1216	22	7781	5	186.3	150	24.37	-15.94
FiLIP-1280	28	8191	6	146.8	630	26.59	-5.11

Table 4. Performances for minimal FHE parameters.

Acknowledgments. Pierrick Méaux is funded by a FSE Incoming Post-Doc Fellowship of the Université catholique de Louvain. François-Xavier Standaert is a Senior

Associate Researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). Work funded in parts by the CHIST-ERA project SECODE and the ERC Project SWORD.

References

1. Albrecht, M.R.: On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part II. LNCS, vol. 10211, pp. 103–129. Springer, Heidelberg (May 2017)
2. Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 430–454. Springer, Heidelberg (Apr 2015)
3. Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. IACR Cryptology ePrint Archive p. 687 (2016)
4. Applebaum, B.: Pseudorandom generators with long stretch and low locality from random local one-way functions. In: Karloff, H.J., Pitassi, T. (eds.) 44th ACM STOC. pp. 805–816. ACM Press (May 2012)
5. Applebaum, B.: Cryptographic hardness of random local functions-survey. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, p. 599. Springer, Heidelberg (Mar 2013)
6. Applebaum, B., Lovett, S.: Algebraic attacks against random local functions and their countermeasures. In: Wichs, D., Mansour, Y. (eds.) 48th ACM STOC. ACM Press (Jun 2016)
7. Armknecht, F., Carlet, C., Gaborit, P., Künzli, S., Meier, W., Ruatta, O.: Efficient computation of algebraic immunity for algebraic and fast algebraic attacks. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004. Springer, Heidelberg (May / Jun 2006)
8. Bellare, M., Yee, B.S.: Forward-security in private-key cryptography. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 1–18. Springer, Heidelberg (Apr 2003)
9. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Goldwasser, S. (ed.) ITCS 2012. pp. 309–325. ACM (Jan 2012)
10. Canteaut, A., Carpov, S., Fontaine, C., Lepoint, T., Naya-Plasencia, M., Paillier, P., Sirdey, R.: Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783. Springer, Heidelberg (Mar 2016)
11. Carlet, C.: Boolean Functions for Cryptography and Error-Correcting Codes, p. 257–397. Encyclopedia of Mathematics and its Applications, Cambridge University Press (2010)
12. Carlet, C., Méaux, P., Rotella, Y.: Boolean functions with restricted input and their robustness; application to the FLIP cipher. IACR Trans. Symmetric Cryptol. (3) (2017)
13. Cogliati, B., Tanguy, T.: Multi-user security bound for filter permutators in the random oracle model. Designs, Codes and Cryptography (09 2018)
14. Courtois, N.: Fast algebraic attacks on stream ciphers with linear feedback. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 176–194. Springer, Heidelberg (Aug 2003)
15. Courtois, N.: Higher order correlation attacks, XL algorithm and cryptanalysis of toyocrypt. In: Lee, P.J., Lim, C.H. (eds.) ICISC 02. LNCS, vol. 2587. Springer, Heidelberg (Nov 2003)
16. Courtois, N., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656. Springer, Heidelberg (May 2003)
17. Couteau, G., Dupin, A., Méaux, P., Rossi, M., Rotella, Y.: On the concrete security of goldreich's pseudorandom generator. In: ASIACRYPT 2018, Part I (2018)
18. Cryan, M., Miltersen, P.B.: On pseudorandom generators in nc 0. In: International Symposium on Mathematical Foundations of Computer Science. Springer (2001)
19. Dobraunig, C., Eichlseder, M., Grassi, L., Lallemand, V., Leander, G., List, E., Mendel, F., Rechberger, C.: Rasta: A cipher with low anddepth and few ands per bit. In: CRYPTO 2018. pp. 662–692 (2018)

20. Duval, S., Lallemand, V., Rotella, Y.: Cryptanalysis of the FLIP family of stream ciphers. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 457–475. Springer, Heidelberg (Aug 2016)
21. Faugère, J.C.: A new efficient algorithm for computing groebner bases. Journal of Pure and Applied Algebra pp. 61–88 (june 1999)
22. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st ACM STOC. pp. 169–178. ACM Press (May / Jun 2009)
23. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (Aug 2013)
24. Gérard, B., Grosso, V., Naya-Plasencia, M., Standaert, F.: Block ciphers that are easier to mask: How far can we go? In: Bertoni, G., Coron, J. (eds.) CHES 2013. Lecture Notes in Computer Science, vol. 8086. Springer (2013)
25. Goldreich, O.: Candidate one-way functions based on expander graphs. Electronic Colloquium on Computational Complexity (ECCC) **7**(90) (2000)
26. Grassi, L., Rechberger, C., Rotaru, D., Scholl, P., Smart, N.P.: Mpc-friendly symmetric key primitives. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) ACM SIGSAC Conference on Computer and Communications Security (2016)
27. Grosso, V., Leurent, G., Standaert, F., Varici, K.: Ls-designs: Bitslice encryption for efficient masked software implementations. In: Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers (2014)
28. Halevi, S., Shoup, V.: Algorithms in HElib. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 554–571. Springer, Heidelberg (Aug 2014)
29. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Cryptography with constant computational overhead. In: 40th ACM STOC. pp. 433–442. ACM Press (May 2008)
30. Knuth, D.E.: Seminumerical Algorithms, The Art of Computer Programming, vol. 2. Addison-Wesley Professional, third edn. (November 1997)
31. Lauter, K., Naehrig, M., Vaikuntanathan, V.: Can homomorphic encryption be practical? Cryptology ePrint Archive, Report 2011/405 (2011)
32. Lin, H., Tessaro, S.: Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. Cryptology ePrint Archive, Report 2017/250 (2017)
33. Lin, H., Vaikuntanathan, V.: Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In: 57th FOCS. pp. 11–20 (2016)
34. Maitra, S., Mandal, B., Martinsen, T., Roy, D., Stănică, P.: Tools in analyzing linear approximation for boolean functions related to flip. In: Chakraborty, D., Iwata, T. (eds.) INDOCRYPT 2018. pp. 282–303 (2018)
35. Méaux, P., Carlet, C., Journault, A., Standaert, F.: Improved filter permutators: Combining symmetric encryption design, boolean functions, low complexity cryptography, and homomorphic encryption, for private delegation of computations. Cryptology ePrint Archive, Report 2019/483 (2019)
36. Méaux, P., Journault, A., Standaert, F.X., Carlet, C.: Towards stream ciphers for efficient FHE with low-noise ciphertexts. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 311–343. Springer, Heidelberg (May 2016)
37. Mesnager, S.: On the nonlinearity of boolean functions with restricted input. Talk at The 13th International Conference on Finite Fields and their Applications (2017)
38. Mesnager, S., Zhou, Z., Ding, C.: On the nonlinearity of boolean functions with restricted input. Cryptography and Communications (Mar 2018)
39. Mossel, E., Shpilka, A., Trevisan, L.: On e-biased generators in NC0. In: 44th FOCS. pp. 136–145. IEEE Computer Society Press (Oct 2003)
40. Piret, G., Roche, T., Carlet, C.: PICARO - A block cipher allowing efficient higher-order side-channel resistance. In: ACNS 2012, Singapore, 2012. Proceedings. pp. 311–328 (2012)