Somewhat/Fully Homomorphic Encryption: implementation progresses and challenges

Guillaume Bonnoron[1,2], Caroline Fontaine[2], Guy Gogniat[3], Vincent Herbert[2,4], Vianney Lapôtre[3], Vincent Migliore[3], and Adeline Roux-Langlois[5]

[1] Chair of Naval Cyber Defense, Ecole Navale - CC600, F-29240 Brest Cedex 9, France,
guillaume.bonnoron@ecole-navale.fr,
[2] CNRS and IMT Atlantique, UMR 6285, Lab-STICC, CS 83818, F-29238 Brest cedex 3, France,
caroline.fontaine@imt-atlantique.fr,
[3] Univ. Bretagne-Sud, UMR 6285, Lab-STICC, F-56100 Lorient, France,
firstname.lastname@univ-ubs.fr,
[4] CEA LIST, Point Courrier 172, 91191 Gif-sur-Yvette Cedex, France,
vincent.herbert@cea.fr,
[5] CNRS - IRISA, Campus universitaire de Beaulieu 35042 Rennes, France,
adeline.roux-langlois@irisa.fr

Abstract. The proposed article aims, for readers, to learn about the existing efforts to secure and implement Somewhat/Fully Homomorphic Encryption ((S/F)HE) schemes and the problems to be tackled in order to progress toward their adoption. For that purpose, the article provides, at first, a brief introduction regarding (S/F)HE. Then, it focuses on some practical issues related to the adoption of (S/F)HE schemes, i.e. the security parameters, the existing implementations and their limitations, and the management of the huge complexity caused by homomorphic calculation. These issues are analyzed with the help of recent related work published in the literature, and with the experience gained by the authors through their experiments.

Keywords: Homomorphic Encryption, Data Privacy, Confidentiality, Security, Real World.

1 Introduction

Homomorphic Encryption (HE) is a recent promising tool in modern cryptography, that allows to carry out operations on encrypted data. The key idea is that performing some operations on encrypted data will provide the same result after decryption as if the computation would have been performed on the original plain data. Then, with such a tool one could outsource storage and/or computation without endangering data's privacy. Figure 1 illustrates different client/server scenarii benefiting from homomorphic encryption. Some examples of applications can be found in the literature, *e.g.* [NLV11,GLN12,LLN14,BPB09,CGGI16b]. While usual cryptographic schemes sometimes have homomorphic properties, for addition [Pai99] or multiplication [ElG85] operations (see also [FG07] for a survey on such partially homomorphic schemes), an important breakthrough has been made in 2009 according to the work of Gentry [Gen09b,Gen09a]. Based on hard lattice problems, he proposed the first *Fully Homomorphic Encryption* (FHE) scheme, enabling to perform an unlimited number of additions and multiplications secretly. Unfortunately, this scheme was too complex to be used in practice. Nevertheless, it introduced an interesting structure as well as a nice trick called *bootstrapping* to reduce the inherent noise that accompany the running of additions and multiplications. Following this seminal work, several FHE have been proposed, but none of them were usable in practice. It is interesting to notice that some FHE are related to more practical schemes called *Somewhat Homomorphic Encryption (SHE) schemes*, that enable an arbitrary number of additions but a bounded number of multiplications. In fact, with the help of bootstrapping, one can design FHE schemes from SHE schemes. Nevertheless, this bootstrapping step adds an extra cost to an already quite heavy process.

To address a particular use case, one must have in mind several constraints that will be crucial for choosing the right homomorphic encryption solution. First, using (S/F)HE schemes will lead to a huge ciphertext expansion (say from 2.000 to 500.000 or even 1.000.000 according to the

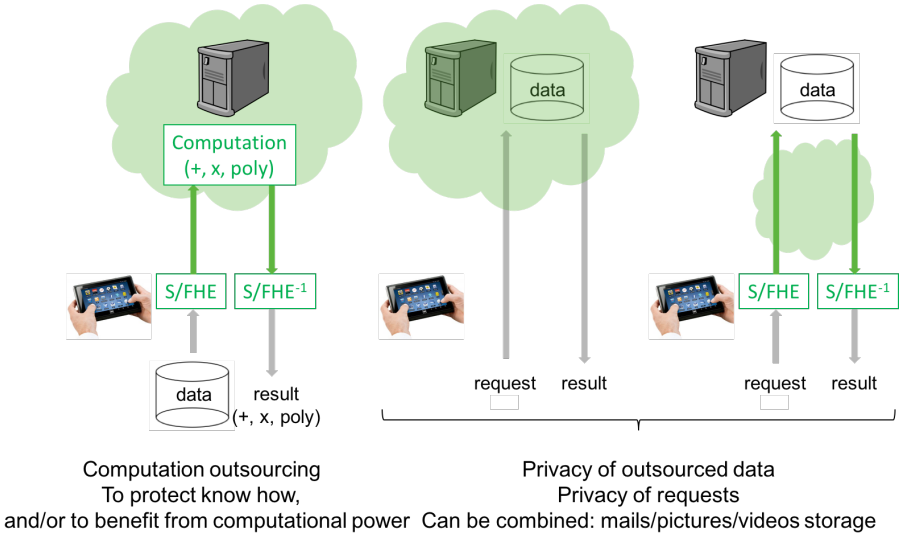

Fig. 1. A need for processing encrypted data, to ensure privacy in outsourced computation, outsourced storage and databases requests. Green areas show what is encrypted.

scheme and the targeted security level). This is due to the fact that homomorphic schemes must be probabilistic to ensure semantic security, and to the particular underlying mathematical structures. Second, as we will see later in this paper, we will need to consider only worse case complexity for the algorithms that will bu run on the encrypted data; also considering that the underlying operations are intrinsically expansive, this will drastically penalize the global running time. These are the most important constraints we have to address. The underlying common point behind these remarks concerns the targeted security level, as it determines the parameters used to use the mathematical appropriate structure that will enable a coherent computation, and then the ciphertext expansion and the running time. Another important parameter concerns the strategy of the developer in terms of flexibility.

Hence, two strategies can be followed to drive the choice. On one hand one can fix a maximum multiplicative depth for the Boolean circuit to be evaluated on the encrypted data. This may take into account a small or at least bounded flexibility for future modifications of the circuit to be evaluated. In this case, SHE is the best choice. On the other hand one may want to be able to use any Boolean circuit and then to handle an unbounded multiplicative depth. In this case, FHE is the only choice.

In this paper, we will focus on SHE schemes because they are the most promising today, and we will discuss their implementations issues. We will also provide some information on the state-of-the-art concerning the use of bootstrapping to modify these schemes intro FHE ones. Our goal is to provide the reader the best starting points to handle the complexity of the issues to address and the efforts made to make these schemes become sufficiently efficient to be used in practice.

2 Existing (F/S)HE solutions

Due to lack of space, we do not provide any precise description and let the reader refer to the mentioned papers to get all the mathematical details related to the schemes.

2.1 SHE from classical crypto world

The first scheme that enabled to perform both additions and multiplications is due to Boneh *et al.* [BGN05]. This pairing-based SHE enables to perform as many additions as wanted, but only one multiplication. Hence, it is not really flexible.

We have worked recently [HF17] on the design of a variant of this scheme that allows to handle multiplicative depth 2 (instead of 1). Our solution employs together two improvements of the original scheme, based on [Fre10,CF15]. We will refer to it as BGN2 in the rest of the document.

2.2 Lattice-based (S/F)HE

Whereas the first FHE scheme has been proposed by Gentry in [Gen09a], the first SHE based on lattices has been propose by Aguilar *et al.* in [AMGH10]. These schemes have been followed by many others. First generation encompass the older ones, like [vDGHV10,SV10,GHS12a,GH11]. Second generation [BGV12,CNT12,BGV12,Bra12,FV12] started with leveled SHE schemes based on modulus switching, which have then been improved to remain scale invariant, etc. They were followed by third generation schemes like [GSW13,BLLN13,BV14,KGV15,DS16]. In fact several of these schemes consist of improvement of previous ones, and the genealogy in rich of cross-fertilization. For example, SHIELD [GSW13] and F-NTRU [DS16] are the third generation schemes equivalent to, respectively, FV [FV12] and YASHE' [BLLN13]. With larger costs for the first homomorphic multiplications, these schemes have a much better asymptotic behavior.

3 Customization and optimization of both program and data

The issues discussed in this section have been addressed in few papers only [AMFF$^+$13,FSF$^+$13,CDS15] for the moment, but they are really important to handle real deployment of this technology.

3.1 Program management

To establish a proper link between the program we want to run and (S/F)HE schemes, one must rely on the fact that any program can be expressed as a Boolean circuit involving `XOR` and `AND` gates, and that `XOR` and `AND` are precisely the addition and multiplication operators over bits. This being said, the game is to express the program through such a Boolean circuit, and to optimize it. The last step will then be to evaluate this optimized Boolean circuit over the encrypted data while replacing `XOR` and `AND` gates by the corresponding encrypted operators provided by a (S/F)HE library.

The optimization step of the Boolean circuit is crucial, as it will give us the multiplicative depth we will have to handle with the (S/F)HE scheme. This will have an impact on the parameters of this scheme (*e.g.* size of the lattice, modulus, etc), and then on the ciphertexts size and on the resulting computation time. Usally, we focus on this multiplicative depth, but we also have to be careful that a very large amount of additions may also lead to heavy computations. Also, sometimes additions following multiplications may have a different impact than additions occurring at lower multiplicative depth. Also, when using leveled (S/F)HE using modulus switching techniques like [BGV12], one has to optimize at which level each ciphertext stands, to avoid any extra and costly modulus switching. At last, the noise growth is usually symmetric between the left and right operands, but for some particular recent schemes like [GSW13] the noise growth is asymmetric over the multiplications. In this case, it is important to optimize which operand will be left or right. All these aspects should be taken into account. This optimization issue has not been sufficiently addressed in the literature, and a lot of work remains to be done to set theoretical bounds and practical strategies to handle such an optimality.

Another issue related with the customization of the Boolean circuit is that it may contain `if-then-else` or `repeat...until` expressions leading to branches with dynamic size depending on the processed data. To handle such statements properly, on must have in mind that the processed data they are depending on are encrypted. Moreover, it is crucial that during the running process no information about the real value of the underlying data may leak. To handle this is easy but costly. In fact, the best approach is to rewrite such parts of the program with the help of Boolean expressions. If we look at the `if-then-else` example, one can rewrite `if c then x=a else x=b` as `x = (c AND a) XOR ((NOT c) AND b)`. The whole expression will then be evaluated over the encrypted data, and the final encrypted result will be the good one without revealing anything on the plain value of `c`. Unfortunately the price to be paid is high, as we always have to evaluated the whole expression, meaning that we always need to evaluate the deepest branch of the tree. Hence, computing over encrypted data always requires the worse-case complexity. This means that when choosing an algorithm to perform a particular computation over encrypted data, one must choose the algorithm that provides the best worse-case complexity (whereas usually we look for the best average-case complexity).

Finally, as (S/F)HE remain costly today, on must be really careful not to perform non necessary heavy computations. According to a particular applicative scenario, one must try to perform as much classical encryption as possible, and to think of some pre-processing over the plaintexts that may help to enlight the computation over the ciphertexts.

3.2 Data management

Some schemes work only over integers, and some over bits. And some can manage both. If the application scenario does not require data depend behavior, then working over integers is usually the best choice. But if the application scenario requires some `if-then-else` or `repeat...until` like statements, then we need to work at the bit level to be able to perform `<`, `>` and `=` operators. In this case, we have to provide home maid basic operators for integers additions and multiplications, floats management,... but this is the price to pay to handle such dynamic behaviors properly.

Some schemes may also be compliant with batching ability, then enabling to group several pieces of data on the same plaintext, typically a polynomial, and to process all these pieces of data at the same time. This drastically reduces the costs in terms of space (memory) and running time, as several plaintexts (resp. ciphertexts) are processed in one shot . In the literature, batching is usually addressed through SIMD and RNS, see for example [SV14,BEHZ16].

4 Flexibility of (S/F)HE schemes in terms of multiplicative depth management

4.1 SHE from classical crypto world

SHE schemes coming from classical crypto world are not flexible. Boneh-Goh-Nissim [BGN05] scheme only enables the evaluation only of degree 2 polynomials, whereas our extension BGN2 enables the evaluation of degree 4 polynomials [HF17].

4.2 Lattice-based (S/F)HE schemes

SHE schemes based on these lattice problems are generally much more flexible and can be turned into fully homomorphic schemes allowing computation with arbitrary depth. The downside for this flexibility is the increased size of ciphertexts and keys, leading to heavier computations. For these schemes, one have to choose the maximum multiplicative depth we want to be able to handle before deployment, as this multiplicative depth will drive the parameters determining the underlying lattice. Once the scheme is deployed, it cannot be modified. Moreover, as we have to handle "the higher the multiplicative depth, the higher the memory and time costs" the parameters much be chosen very carefully to maintain an acceptable cost while ensuring sufficient flexibility for future process.

FHE schemes are generally heavier to deploy, as they need the bootstrapping step to be sure to enhance the noise growth properly whatever the multiplicative depth. Hence, they should be used only if one does not know in advance the multiplicative depth we would like to handle in the future in the targeted application scenario. Good recent works to better understand the limits of this solution are [PV15], which provides a way to optimize the bootstraps management (for any FHE), and [CGGI16a] that proposes the more efficient current way to execute bootstrapping (especially for FHE based on [GSW13]).

5 Security of (S/F)HE schemes

By nature, these schemes have been designed to enhance privacy and security. Hence, the analysis of their security is crucial. The best security level that has been achieved for such schemes is IND-CPA. We know that IND-CCA2 in not achievable, and the design of efficient schemes achieving IND-CCA1 is still open. Now, we will split the security study according to the underlying mathematical rationales.

5.1 SHE from classical crypto world

Schemes which are based on mathematical objects which have already been deeply studied by the cryptographic community are better understood, and their security is easier to manage. Among such schemes, one can mention partially homomorphic schemes as ElGamal, Paillier, and their variants, which are out of the scope of this article as they cannot handle at the same time additions and multiplications. As mentioned in Section 2.2, one can also mention Boneh-Goh-Nissim scheme [BGN05], a pairing-based SHE scheme which enables to perform as many additions as wanted, but only one multiplication, and our scheme called BGN2 which can handle one more multiplication depth [HF17].

The security of BGN2 is based on the generalized subgroup decision assumption[6]. This problem is derived from the decision Diffie-Hellman assumption [Bon98]. Two possible choices to instantiate groups are to select either an elliptic curve or an hyperelliptic curve. We place ourselves in the first case, where the security assumption reduces to the elliptic curve discrete logarithm problem. The recommended group size is given by different academic and private organizations at www.keylength.com according standard security levels. The order of ciphertext expansion in BGN2 is thousands.

5.2 Lattice-based (S/F)HE

The *Learning With Errors* problem is about solving a system of several *noisy* linear equations. Its ring variant, *Ring-LWE*, allows to design more efficient schemes with faster computations and smaller keys and ciphertexts. These problems attract large attention, beyond HE, because they are believed to be quantum resistant: no known quantum algorithms perform better than the classical ones against them.

These schemes were introduced in the previous decade, i.e. quite recently in the time scale of cryptography, and one must tell that there is still a gap between the theoretical hardness proofs and the practical behavior of the known attacks.

Formal proofs The first FHE scheme of Gentry [Gen09b] was based on two assumptions, the *Bounded Distance Decoding problem* and the *Sparse Subset Sum problem* which are not standard in lattice-based cryptography. The first scheme proven secure under the LWE assumption is proposed by Brakerski and Vaikuntanathan in 2011 [BV11]. This result was followed by numerous constructions based on LWE and Ring-LWE (as [Bra12,BGV12]...).

The LWE and Ring-LWE problem are proven to be at least as hard as well-known hard problems (in their worst-case) on lattices (respectively on ideal-lattices). Another advantage of recent lattice-based schemes is that their security is proven under those assumptions.

Experimental security evaluation Even at this point, concrete security behind homomorphic encryption is still moving. Thus, extracting realistic parameters for a given security level is a main challenge of (S/F)HE scenario. The standard approach so far is to focus on concrete attack means. Building on surveys about existing attacks against LWE [APS15] and Ring-LWE [Pei16], experiments must be pushed further to provide experimental results of secured parameters for most promising homomorphic schemes.

Among these most promising schemes one can mention YASHE' [BLLN13], FV [FV12] and SHIELD [GSW13]. YASHE' and FV have been published almost at the same time, but YASHE' took benefit from a strong lobby and became more popular in the proposed implementations. However, confidence in this scheme has been recently damaged by the subfield/sublattice attack [ABD16,KF16]. Making YASHE' immune to these attacks would lead to oversize its parameters, far too much for practical use [DGBL$^+$15]. This is why among second generation schemes FV is now the real challenger, and has received a lot of attention during the past months [LCP16,Cry16,BEHZ16].

[6] We choose to employ asymetric pairings to compute homomomorphic product of fresh ciphertexts. The use of symmetric pairings would change the computational hardness assumption[Fre10, page 46].

6 Existing implementations of (S/F)HE schemes

6.1 Software implementations

We implemented BGN2, which have been presented and briefly discussed above. In this cryptosystem, the homomorphic multiplication asks to compute pairings. We chose to compute an optimal Ate pairing over an elliptic curve in the Barreto-Naehrig curve family [BN05] using a program called DCLXVI [NNS10]. This work is not yet published, but will be available soon both as an article and as a software.

Concerning lattice-based (S/F)HE schemes, several implementations have emerged since their introduction in 2009. Due to the pace of evolution in the theoretical field, some are now outdated but served as good proof of concept in the early days of (S/F)HE ([Bre,Cor,Lep]). Other libraries ([DGBL$^+$15,Hal,Cry16,LCP16]) implement the latest techniques described in [SV10,FV12,BGV12,BLLN13]. There are also private implementations like those used in [AMFF$^+$13,FSF$^+$13,CDS15,BEHZ16]. Most of the libraries aim at providing tools for experiments to the academic community, except [DGBL$^+$15,LCP16,CDS15] which can be used as building blocks for more industrialized developments.

Today, no complete benchmark is available to provide a fair and complete overviews of the efficiency of all these schemes. The reader can refer to [LN14] for a comparison of FV and YASHE', and to [MBF16] for a first discussion and comparison of FV, SHIELD and F-NTRU in terms of pros and cons, and parameters setting.

6.2 Hardware implementations

Hardware implementation is one of the two principal ways to accelerate FHE/SHE schemes with dedicated components. The second one is the GPU acceleration. Even if performances using GPU are very scheme-dependent, it can be a good alternative to set up an homomorphic server quickly with acceptable performances [KGV15].
Hardware accelerators focus on accelerating the most complex operation of Homomorphic Encryption Schemes, the multiplication of homomorphic operands. Depending on the scheme, a million-bit integer multiplier or a polynomial of degree $n \in [4096, 32768]$ with coefficients of size $\log_2 q \in [125, 1228]$ is required. In [DOS13], an ASIC implementation of million-bits multiplier performs the multiplication operation in 7.74 ms using NTT algorithm. This computation time corresponds to the computation time on a Xeon, but can be implemented as a co-processor and thus requiring a much smaller area. For polynomial based homomorphic encryption, due to the fact that one must address various size of polynomials, different architectures have been investigated. To our knowledge, all implementations are based on NTT algorithm too, except in [MMRL$^+$17] for small size polynomials which implements Karatsuba algorithm instead. In [PNPM], a usual but optimized NTT implementation is presented for two parameter sets. The proposed accelerator performs an homomorphic multiplication in 6.5 ms for $n = 4096$ and $\log_2 q = 125$ bits, and 48 ms for parameters $n = 16384$ and $\log_2 q = 512$ bits. Authors of [PNPM] implemented 512×512 bits multipliers with a small modular reduction by selecting a Solinas prime modulus [Sol11]. Due to the size of polynomials and coefficients, a cache is implemented to connect the external memory used to store intermediate coefficients. They also reported a bottleneck due to the divide and rounding required by YASHE' scheme, especially for large integers. That is why in [SRJV$^+$] a precomputation is performed on polynomials to reduce the size of coefficients. They split a ciphertext into a few polynomials by using the *Chinese Reminder Theorem* (CRT) on each coefficient. The overall architecture is based on an array of crypto-units, which gives some flexibility to process several residue polynomials in parallel. For parameters $n = 32768$ and $\log_2 q = 1228$ bits, their accelerator performs an homomorphic multiplication in 121 ms including 25 ms spent for CRT.
Table 1 summarize the different hardware implementations available for both integer based and polynomial based Homomorphic Encryption Schemes.

7 How to handle such a huge complexity and expansion?

(S/F)HE schemes defined on Elliptic Curves and Pairing present a smaller complexity and expansion, and their security level is quite clear, but are very limited in terms of multiplicative depth.

Table 1. Timing results for the hardware implementation of Homomorphic Encryption.

Integer based Homomorphic encryption					
Scheme	Algorithm	Size	Homomorphic Encryption	Homomorphic Multiplication	Work
Gentry-Halevy	NTT	1 M bits	2.09 s	7.74 ms	[DOS13]
DHGV		19 M bits	3.9 s	*no results*	[CMO+]

Polynomial based Homomorphic encryption						
Scheme	Algorithm	n	$\log_2 q$	Homomorphic Encryption	Homomorphic Multiplication	Work
YASHE'	Karatsuba	2560	124 bits	*not implemented*	4.73 ms	[MMRL+17]
	NTT	4096	125 bits		6.5 ms	[PNPM]
		16384	512 bits		48 ms	
		32768	1228 bits		121 ms	[SRJV+]

For some applications this may be sufficient, and particularly interesting, this is why we discussed them here.

Nevertheless, the biggest hope for the future comes from lattice-based schemes, which promise to handle larger multiplicative depths processing. Once their security will be better understood, their main drawbacks are their algorithmic complexity and the related ciphertext expansion. This is why it is important to pursue designing new schemes and to look for lighter solutions. Current ciphertext expansions can go from $10,000$ up to $1,000,000$, depending on the schemes and on the parameters that have been chosen (and which are directly related with the targeted security level). The encrypted data must be uploaded from the client device to the server, then processed on the server, and finally the encrypted result must be downloaded from the server to the client device. Hence, its size is critical.

To reduce the size of the uploaded data on the first step, [NLV11] proposed to combine the (S/F)HE scheme with symmetric encryption schemes. The main idea is that the data to be uploaded will be encrypted by the symmetric encryption scheme, and then sent to the server without any size expansion. Hence, the server will trans-crypt this encrypted data to produce a new ciphertext which corresponds to the encryption of the same data with the (S/F)HE that will be used on the server to perform the desired computation. This trans-cryption step will require the decryption circuit of the underlying symmetric encryption scheme to be evaluated by the (S/F)HE scheme. This step's complexity is critical, and will lead the choice of the symmetric cipher to use. Following this idea, several symmetric encryption schemes have been investigated. We will first mention on-the-shelf block ciphers like AES [GHS12b,CCK+13,DHS14], and the lightweight block ciphers Simon [LN14] and Prince [DSES14]. But the evaluation of these ciphers by the (S/F)HE remains too complex, and recent papers proposed new ciphers designed specifically for this purpose (*i.e.* with a low multiplicative depth): the block cipher Low-MC [ARS+15], which has been broken [DLMW15] and patched [Rec16]; the stream cipher Kreyvium [CCF+16], whose security is the same as the well-studied stream cipher Trivium; and a more recent stream cipher proposal called FLIP [MJSC16], which should be used carefully [DLR16]. These papers include experimental material and results. More exotic solutions are discussed in [FHK16], but without experimental data in the paper.

The second way to reduce ciphertexts weight is to pack several inputs on the same plaintext structure through batching. This has been briefly discussed in Section 3.2

8 Acknowledgement

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 643964.

9 Conclusion

There is still a lot of work to be done, but everything is moving fast and recent progress are quite impressive. Hence, for some applications which are not too critical in terms of memory and time

costs it is time to adopt a practical approach to make dream become reality. How to choose the good (S/F)HE scheme for a given application scenario, and how to set it up in the best way? The answer is not trivial at all, and this article provides some hints concerning the implementation issues of these promising but still tricky and heavy schemes. Our goal was to share our discussions and reflections about all the identified issues, and to provide ad-hoc references to help the reader in his exploration of a very prolific and dense literature. It is clear that more comparisons should be performed between the most promising schemes. A few papers compared two schemes at a time, like [LN14], and a first attempt to provide a wider analysis can be found in [MBF16]. But it is clear that it should be pushed further, and that fair benchmarks based on available implementations have to be driven. Moreover, a fair and precise comparison should also be driven to properly compare SHE schemes coming from classical crypto world with lattice-based ones when targeting a small multiplicative depth, in terms of time and space complexity. At the same time, open issues remain concerning a precise evaluation of the practical security of lattice-based schemes, as well as on the optimization of the Boolean circuits we want to evaluate over the encrypted data.

References

ABD16. M. Albrecht, S. Bai, and L. Ducas. A subfield lattice attack on overstretched ntru assumptions: Cryptanalysis of some fhe and graded encoding schemes. Cryptology ePrint Archive, Report 2016/127, 2016.

AMFF+13. Carlos Aguilar-Melchor, Simon Fau, Caroline Fontaine, Guy Gogniat, and Renaud Sirdey. Recent advances in homomorphic encryption: A possible future for signal processing in the encrypted domain. *IEEE Signal Processing Magazine*, 30(2):108–117, 2013.

AMGH10. C. Aguilar Melchor, P. Gaborit, and J. Herranz. Additively homomorphic encryption with d-operand multiplications. In *Annual Cryptology Conference*, pages 138–154. Springer, 2010.

APS15. Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

ARS+15. Martin Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In *EUROCRYPT, Part I*, volume 9056 of *LNCS*, pages 430–454. Springer, 2015.

BEHZ16. Jean-Claude Bajard, Julien Eynard, Anwar Hasan, and Vincent Zucca. A Full RNS Variant of FV like Somewhat Homomorphic Encryption Schemes. Cryptology ePrint Archive, Report 2016/510, 2016. `http://eprint.iacr.org/2016/510`.

BGN05. 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.

BGV12. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proc. of the 3rd Innovations in Theoretical Computer Science Conference – ITCS 2012*, pages 309–325. ACM, 2012.

BLLN13. J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In *Proc. of Cryptography and Coding: 14th IMA International Conference – IMACC 2013*, pages 45–64. Springer, 2013.

BN05. 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.

Bon98. 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.

BPB09. 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.

Bra12. Z. Brakerski. Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP. In *Proc. of CRYPTO*, volume 7417 of *LNCS*, pages 868–886. Springer, 2012.

Bre. Michael Brenner. Hcrypt project. Available at `http://www.hcrypt.com`.

BV11. Z. Brakerski and V. Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. In *Proc. of FOCS*, pages 97–106, 2011.

BV14. Z. Brakerski and V. 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.

CCF+16. Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrede Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. Stream ciphers: A Practical Solution for Efficient Homomorphic-Ciphertext Compression. In *Fast Software Encryption 2016*, Fast Software Encryption 2016, Bochum, Germany, March 2016. Springer Verlag.

CCK+13. Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrède Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch Fully Homomorphic Encryption over the Integers. In *EUROCRYPT*, volume 7881 of *LNCS*, pages 315–335. Springer, 2013.

CDS15. Sergiu Carpov, Paul Dubrulle, and Renaud Sirdey. Armadillo: a compilation chain for privacy preserving applications. In *Proceedings of the 3rd International Workshop on Security in Cloud Computing*, pages 13–19. ACM, 2015.

CF15. 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.

CGGI16a. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. *Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds*, pages 3–33. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

CGGI16b. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. A homomorphic LWE based e-voting scheme. In *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*, pages 245–265, 2016.

CMO+. Xiaolin Cao, Ciara Moore, Maire O'Neill, Neil Hanley, and Elizabeth O'Sullivan. High-Speed Fully Homomorphic Encryption over the Integers. In *Proc. of the Workshop on Encrypted Computing and Applied Homomorphic Cryptography – WAHC 2014*.

CNT12. 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.

Cor. Coron, Jean-Sébastien. An implementation of the DGHV fully homomorphic scheme. Available at `https://github.com/coron/fhe`.

Cry16. CryptoExperts. FV-NFLlib. Available at `https://github.com/CryptoExperts/FV-NFLlib`, 2016.

DGBL+15. Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Manual for Using Homomorphic Encryption for Bioinformatics. Technical Report MSR-TR-2015-87, November 2015.

DHS14. Yarkin Doröz, Yin Hu, and Berk Sunar. Homomorphic AES Evaluation using NTRU. *IACR Cryptology ePrint Archive*, 2014:39, 2014.

DLMW15. Itai Dinur, Yunwen Liu, Willi Meier, and Qingju Wang. Optimized Interpolation Attacks on LowMC. *IACR Cryptology ePrint Archive*, 2015:418, 2015.

DLR16. Sébastien Duval, Virginie Lallemand, and Yann Rotella. Cryptanalysis of the FLIP Family of Stream Ciphers. *IACR Cryptology ePrint Archive*, (271), 2016.

DOS13. Yakin Doroz, E Ozturk, and Berk Sunar. Evaluating the Hardware Performance of a Million-Bit Multiplier. In *Proc. of Euromicro Conference on Digital System Design – DSD 2013*, 2013.

DS16. Y. Doröz and B. Sunar. Flattening ntru for evaluation key free homomorphic encryption. Cryptology ePrint Archive, Report 2016/315, 2016.

DSES14. Yarkin Doröz, Aria Shahverdi, Thomas Eisenbarth, and Berk Sunar. Toward Practical Homomorphic Evaluation of Block Ciphers Using Prince. In *WAHC*, volume 8438 of *LNCS*, pages 208–220. Springer, 2014.

ElG85. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

FG07. C. Fontaine and F. Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP J. Inf. Secur.*, 2007(1):1–15, 2007.

FHK16. Pierre-Alain Fouque, Benjamin Hadjibeyli, and Paul Kirchner. *Homomorphic Evaluation of Lattice-Based Symmetric Encryption Schemes*, pages 269–280. Springer International Publishing, Cham, 2016.

Fre10. 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.

FSF+13. Simon Fau, Renaud Sirdey, Caroline Fontaine, Carlos Aguilar-Melchor, and Guy Gogniat. Towards practical program execution over fully homomorphic encryption schemes. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on*, pages 284–290. IEEE, 2013.

FV12. Junfeng Fan and Frederik Vercauteren. Somewhat Practical Fully Homomorphic Encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.

Gen09a. C. Gentry. *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford University, 2009.

Gen09b. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.

GH11. 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.

GHS12a. 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.

GHS12b. Craig Gentry, Shai Halevi, and Nigel P Smart. Homomorphic evaluation of the AES circuit. In *Advances in Cryptology–CRYPTO 2012*, pages 850–867. Springer, 2012.

GLN12. Thore Graepel, Kristin E. Lauter, and Michael Naehrig. ML Confidential: Machine Learning on Encrypted Data. In *ICISC*, volume 7839 of *LNCS*, pages 1–21. Springer, 2012.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology–CRYPTO 2013*, pages 75–92. Springer, 2013.

Hal. Shai Halevi. HElib. Available at `https://github.com/shaih/HElib`.

HF17. Vincent Herbert and Caroline Fontaine. Software implementation of 2-depth pairing-based homomorphic encryption scheme. Cryptology ePrint Archive, Report 2017/091, 2017. `http://eprint.iacr.org/2017/091`.

KF16. Paul Kirchner and Pierre-Alain Fouque. Comparison between Subfield and Straightforward Attacks on NTRU. Cryptology ePrint Archive, 2016/717, 2016.

KGV15. Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers. *IEEE Transactions on Computers*, PP(99):1–1, 2015.

LCP16. Kim Laine, Hao Chen, and Rachel Player. Simple encrypted arithmetic library - seal (v2.1). Technical report, September 2016.

Lep. Lepoint, Tancrède. A proof-of-concept implementation of the homomorphic evaluation of SIMON using FV and YASHE. Available at `https://github.com/tlepoint/homomorphic-simon`.

LLN14. Kristin Lauter, Adriana López-Alt, and Michael Naehrig. Private Computation on Encrypted Genomic Data. In *LATINCRYPT*, LNCS, 2014.

LN14. Tancrède Lepoint and Michael Naehrig. A Comparison of the Homomorphic Encryption Schemes FV and YASHE. In *AFRICACRYPT*, volume 8469 of *LNCS*, pages 318–335. Springer, 2014.

MBF16. 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.

MJSC16. Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. *Towards Stream Ciphers for Efficient FHE with Low-Noise Ciphertexts*, pages 311–343. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

MMRL+17. Vincent Migliore, Maria Mendez Real, Vianney Lapotre, Arnaud Tisserand, Caroline Fontaine, and Guy Gogniat. Hardware/Software co-Design of an Accelerator for FV Homomorphic Encryption Scheme using Karatsuba Algorithm. *Accepted to IEEE Transactions on Computers*, 2017.

NLV11. Michael Naehrig, Kristin E. Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *ACM CCSW*, pages 113–124. ACM, 2011.

NNS10. 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.

Pai99. 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.

Pei16. Chris Peikert. How (not) to instantiate Ring-LWE. In *International Conference on Security and Cryptography for Networks*. Springer, 2016.

PNPM. Thomas Pöppelmann, Michael Naehrig, Andrew Putnam, and Adrian Macias. Accelerating Homomorphic Evaluation on Reconfigurable Hardware. In *Proc. of Cryptographic Hardware and Embedded Systems – CHES 2015*, pages 143–163. Springer.

PV15. Marie Paindavoine and Bastien Vialla. Minimizing the number of bootstrappings in fully homomorphic encryption. In *Selected Areas in Cryptography - SAC 2015 - 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers*, pages 25–43, 2015.

Rec16. Christian Rechberger. The FHEMPCZK-Cipher Zoo. Presented at the FSE 2016 rump session, 2016.

Sol11. Jerome A. Solinas. Generalized Mersenne Prime. In *Encyclopedia of Cryptography and Security*, pages 509–510. Springer, 2011.

SRJV$^+$. Sujoy Sinha Roy, Kimmo Järvinen, Frederik Vercauteren, Vassil Dimitrov, and Ingrid Verbauwhede. Modular Hardware Architecture for Somewhat Homomorphic Function Evaluation. In *Proc. of Cryptographic Hardware and Embedded Systems – CHES 2015*, pages 164–184. Springer.

SV10. 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.

SV14. N. P. Smart and F. Vercauteren. Fully homomorphic simd operations. *Designs, Codes and Cryptography*, 71(1):57–81, 2014.

vDGHV10. 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.