Efficient Methods for Lattice-Based Cryptography

João Marcos de Mattos Barguil[1], **Paulo S. L. M. Barreto**[1]

[1]Department of Computer and Digital Systems Engineering
University of São Paulo, São Paulo, Brazil

jbarguil@usp.br, pbarreto@larc.usp.br

Abstract. *Lattices have been applied in many different ways in cryptography. Firstly used for the destruction of cryptosystems, they were later applied in the construction of new schemes, including asymmetric cryptosystems, blind signature schemes and the first methods for fully homomorphic encryption. Nonetheless, performance is still prohibitively slow in many cases. In this work, we expand techniques originally devised for homomorphic encryption, making them more general and applying them to the GGH-YK-M cryptosystem, a lattice-based public-key cryptosystem, and to the LMSV scheme, the only known homomorphic scheme that has not succumbed to IND-CCA1 key recovery attacks to this date. In our tests, we reduce public key bandwidth occupation of GGH-YK-M by an order of complexity, specifically, from* $O(n^2 \lg n)$ *down to* $O(n \lg n)$ *bits, where* n *is a public parameter of the scheme. The new technique also attains faster processing in all operations involved in an asymmetric cryptosystem, that is, key generation, encryption, and decryption. The most significant improvement in performance is in key generation, which becomes more than 3 orders of magnitude faster than previous results, while encryption becomes about 2 orders of magnitude faster. For decryption, our implementation is ten times faster than the literature. We also show that it is possible to improve security of LMSV against the quantum key recovery attacks recently published by British GCHQ. We do so by adopting non-cyclotomic lattices based on nearly-circulant irreducible polynomial rings. In our implementation, performance of encryption remains virtually the same, and decryption becomes slightly worse, a small price to pay for the improved security. Key generation, however, is much slower, due to the fact that it is necessary to use a more generic and expensive method. The existence of highly efficient dedicated methods for key generation of this secure variant of LMSV remains as an open problem.*

1. Background and motivation

There is a rising, medium-to-long term concern with the potential technological viability of quantum computers, because traditional cryptosystems based on the assumed hardness of integer factorization or discrete logarithm computation can be successfully attacked with the help of this new kind of equipment. New schemes based on different computational problems are thus necessary to address this concern, leading to the development of purely classical, but believed to be quantum-resistant constructions known as *post-quantum* cryptosystems.

One of the most popular families of post-quantum cryptosystems is that of schemes based on the theory of lattices. Lattices have permitted advancements in asym-

metric cryptography [Goldreich et al. 1997] and play a crucial role in the development of homomorphic schemes [Gentry 2009b].

One of the first lattice-based encryption schemes, proposed by Goldreich, Goldwasser and Halevi [Goldreich et al. 1997], or GGH for short, was later broken by Nguyen [Nguyen 1999], who proved that the original scheme had structural flaws. For several years GGH was deemed irretrievably broken, until Yoshino and Kunihiro [Yoshino and Kunihiro 2012] described a variant that prevented all known attacks, named GGH-YK. However, this variant does not allow for the construction of proper parameter sets, and therefore has no practical use. More recently, Barros and Schechter [de Barros and Schechter 2014] expanded this construction, proposing a modification called GGH-YK-M that effectively yields a suitable parametrization. The result is very promising, as it brings the simplicity of GGH and GGH-YK back to life. This family of schemes may, once again, be a viable foundation for the construction of other post-quantum systems.

Another relevant cryptosystem based on GGH-style lattices is the pioneering scheme devised by Gentry [Gentry 2009a]. It is the first known instance of *fully homomorphic encryption*, that is, the first scheme that supports arbitrary arithmetic to be performed over encrypted data while still yielding valid results upon decryption, as if the computation had been done over unencrypted data. It produces, though, very large keys, of the order of Terabytes, and demands Gigabytes of ciphertext to encrypt a single bit. Nevertheless, this work represents a major breakthrough, and prompted the development of many other constructions, some even based on different kinds of lattices. Unfortunately, most of these have already succumbed to cryptanalysis [Szydlo 2003, Chenal and Tang 2014], with the exception of the method proposed by Loftus *et al.* [Loftus et al. 2012], known as LMSV and a direct descendant of Gentry's original scheme. That scheme is *proven* to be IND-CCA1, uniquely among all fully or somewhat homomorphic encryption schemes known today.

Nonetheless, the biggest drawbacks of all cryptosystems based on GGH-style lattices, up to now, are their latent high bandwidth occupation and computational cost, which make this family of schemes less competitive in practice with other lattice-based encryption methods. The technique proposed by Smart and Vercauteren [Smart and Vercauteren 2010] and improved by Gentry and Halevi [Gentry and Halevi 2011], aimed at homomorphic encryption, can be modified to tackle these specific issues, being applicable not only in homomorphic schemes, but also in traditional public-key cryptography.

Very recently, the British Government Communications Headquarters (GCHQ) announced a quantum attack on their homemade Soliloquy scheme [Campbell et al. 2014]. This revelation has caused great discussion, as it allows for the recovery of the private key in polynomially-bounded time complexity. The issue is still surrounded by controversy and far from settled, but it is nevertheless a very important topic and must be taken into account when designing improvements to these methods.

2. Goals

The broad goal of this work is to layout efficient methods for lattice-based cryptography. Our focus is on improving the performance of constructions based on ideal lattices, such as the aforementioned GGH-style cryptosystems. Because efficiency in general is still one of the major hindrances to the development of these families of schemes, our goal is to allow not only for better processing times, but also for lower bandwidth occupation.

We target at generalizing existing methods, expanding their range of applications for contexts other than the ones they were originally designed for. In order to investigate the applicability of such methods in both traditional asymmetric cryptography and homomorphic encryption, we study the specific cases of the GGH-YK-M and LMSV constructions.

Also, it is crucial that these improvements do not affect negatively the security of said schemes. Moreover, increasing their security is also devised as a complimentary goal, taking into account the newly developed quantum key recovery attacks.

3. Contributions

In this work, we extend the technique proposed by Gentry and Halevi, making it more comprehensive, and apply it to a traditional public-key encryption scheme and a homomorphic cryptosystem, specifically, the GGH-YK-M and LMSV schemes. We reduce public key bandwidth occupation of GGH-YK-M by an order of complexity, specifically, from $O(n^2 \lg n)$ down to $O(n \lg n)$ bits, where n is a public parameter of the scheme. The new technique also attains faster processing in all operations involved in an asymmetric cryptosystem, that is, key generation, encryption, and decryption. The most significant improvement in the performance of GGH-YK-M is in key generation, which becomes more than 3 orders of magnitude faster than previous results, while encryption becomes about 2 orders of magnitude faster. For decryption, our implementation is ten times faster than the literature.

In LMSV, our main focus is on increasing its security against the quantum attack devised by GCHQ. We suggest a different choice of polynomial ring, namely, an irreducible instance that yields nearly circulant matrices. This particular choice is able to successfully thwart the new attack and does not have an appreciable impact on the performance of encryption and decryption, if compared to usual ring choices. Key generation, however, is greatly affected, as there is no known dedicated technique and it is necessary to resort to more generic algorithms. How to improve efficiency of key generation in this particular case remains as an open research problem, but our results show that it is possible to increase security of the scheme without great loss of performance in encryption and decryption.

3.1. Improvements on efficiency

We propose a new and more efficient method to compute the Hermite normal form (HNF) [Cohen 1993, Section 2.4.2] of matrices. In GGH-style cryptosystems, the HNF is widely used as the public-key, so improving its calculation directly impacts key generation. The obvious way to obtain shorter keys in other lattice-based settings, namely, resorting to certain rings of structured (e.g., circulant or negacyclic) matrices, fails for

GGH because mapping the private key to a public key, that is, computing the HNF, ends up destroying the underlying structure that would enable the size reduction.

Contrary to this intuitive observation, one can still benefit from an underlying structure in the private key to reduce the size of the public key in a nontrivial way. This was first indicated by Smart and Vercauteren [Smart and Vercauteren 2010], but it seems to require computing the HNF of the lattice basis. Gentry and Halevi [Gentry and Halevi 2011, Lemma 1] offer a proof of this property that avoids computing the HNF for the case $p(x) = x^n + 1$ (where n is a power of 2). We show that, in fact, it holds for any ideal matrix, regardless of the choice of $p(x)$, even though some choices may be more efficient (and possibly more secure) than others.

Furthermore, by applying this method for generating keys, both encryption and decryption can be simplified, allowing for an even greater impact on general performance of these cryptosystems.

3.2. Improvements on security

In recent times, new key recovery attacks to cyclotomic rings have been described. There exist classical approaches, but a new quantum attack in particular has provoked heated discussions, as it represents big news related to what had been always deemed as *"post-quantum cryptography"* up until this point. It is known as the Soliloquy [Campbell et al. 2014] quantum-attack, developed by GCHQ. This issue is still somewhat controversial, in particular whether the attack is polynomial-time or not. There is still very limited literature on the subject, and the matter is far from settled. Nonetheless, it is very important to take note of it, as it seems to apply to some of the most commonly used lattices, including the homomorphic schemes of Smart-Vercauteren [Smart and Vercauteren 2010] and LMSV [Loftus et al. 2012].

The new method mentioned in 3.1 can be modified to improve security against Soliloquy, without great impact on performance. We do so by adopting the suggestion by Bernstein [Bernstein 2014], namely, adopting a field of form $\mathcal{Z}[x]/(x^n - x - 1)$, which yields nearly circulant matrices and fairly efficient arithmetic, instead of circulant fields of form $\mathcal{Z}[x]/(x^n - 1)$.

4. Results

We have implemented the newly developed methods, and report the results obtained by applying them to two cryptosystems: GGH-YK-M and LMSV. For the former, our alternative reduces public key bandwidth occupation by an order of complexity, specifically, from $O(n^2 \lg n)$ down to $O(n \lg n)$, where n is a public parameter of the scheme. The new technique also attains faster processing in all operations involved in a public-key cryptosystem, that is, key generation, encryption, and decryption. By far the most pronounced improvement in performance is in key generation, which becomes more than 3 orders of magnitude faster than published results [de Barros and Schechter 2014], while encryption becomes about 2 orders of magnitude faster. For decryption, our implementation is ten times faster than the literature. Key sizes are essentially the same in our proposal for a given dimension n regardless of the choice of $p(x)$. Sample public key sizes are listed on Table 2.

Table 1. Timings for GGH-YK-M (in seconds)

source	(n, σ, h, k)	keygen (s)	encrypt (ms)	decrypt (ms)
previous[1]	$(350, 256, 526, 64)$	368.16	13.3	37.6
Java	$(353, 256, 526, 64)$	0.34	2.7	55.2
C	$(353, 256, 526, 64)$	0.10	0.1	4.8
previous[1]	$(400, 256, 601, 64)$	692.48	15.5	59.8
Java	$(401, 256, 601, 64)$	0.46	2.0	67.9
C	$(401, 256, 601, 64)$	0.12	0.1	5.3
Java	$(509, 256, 769, 80)$	1.04	4.0	166.3
C	$(509, 256, 769, 80)$	0.22	0.2	9.4
Java	$(512, 256, 769, 80)$	3.01	2.4	124.5
C	$(512, 256, 769, 80)$	0.11	0.2	9.8

[1] [de Barros and Schechter 2014].

Table 2. Public key sizes for GGH-YK-M (in bits)

source	(n, σ, h, k)	$\|pk\|$
previous[1]	$(350, 256, 526, 64)$	1157800
ours	$(353, 256, 526, 64)$	6682
previous[1]	$(400, 256, 601, 64)$	1543200
ours	$(401, 256, 601, 64)$	7738
previous[1]†	$(512, 256, 769, 80)$	2621440
ours	$(512, 256, 769, 80)$	10240

[1] [de Barros and Schechter 2014].

† Inferred.

For LMSV [Loftus et al. 2012], we compare the cyclotomic ring $\mathcal{Z}[x]/(x^n - 1)$ and the more secure irreducible ring $\mathcal{Z}[x]/(x^n - x - 1)$. Performance of encryption remains virtually the same, and decryption becomes slightly worse. Key generation, however, is much slower, due to the fact that it is necessary to use a more generic and expensive method. The scheme was implemented in Java on an Intel i5-2450M 2.5 GHz platform under 64-bit Ubuntu 14.10, and the results are shown in Table 3.

Table 3. Timings for LMSV

$P(x)$	n	keygen (s)	encrypt (ms)	decrypt (ms)
$x^n - 1$	353	0.3	1.5	41.1
$x^n - x - 1$	353	15.3	1.8	45.0
$x^n - 1$	401	0.4	2.5	60.1
$x^n - x - 1$	401	20.3	2.4	64.8
$x^n - 1$	509	1.4	7.2	129.9
$x^n - x - 1$	509	42.8	7.3	134.0
$x^n - 1$	2039	100.3	390.0	10.4×10^3
$x^n - 1$	8191	7.3×10^3	18.3×10^3	879×10^3

Despite being a widely used and throughly available platform, Java provides only limited support for cryptographic applications. The biggest hindrance is due to the fact that *BigInteger* objects are immutable. This means that new instances are created for each

and every arithmetic operation. When a large amount of such operations is executed, a significant impact on performance is observed, caused not by the arithmetic calculations themselves, but by the expensive memory management associated to repeated object instantiation. Analyzing the results presented in Table 3, it is clear that for $n > 509$ these limitations distort the obtained results, as timings increase disproportionately fast.

Implementing the algorithm in C or C++, for instance, could help achieve better performance. It would not, however, address the main issue, that is, the complexity of key generation in $\mathcal{Z}[x]/(x^n - x - 1)$. It is then an open question whether the lower *asymptotic* complexity generates an appreciable improvement for the rather small values of n that are used in practice. Hence, the issue of poor performance in $\mathcal{Z}[x]/(x^n - x - 1)$ is not simply a matter of fine-tuning the implementation, but still an algebraic problem.

5. Publications

Part of the work presented in Section 4 and some preliminary results have been published [Barguil et al. 2014] at SBSeg 2014.

While exploring applications of somewhat homomorphic encryption that might benefit from our new techniques, we briefly considered some e-cash protocols, in particular Sarkar's [Sarkar 2013] because of its apparent reliance on ring operations. While that protocol turned out not to be a significant example of a scenario where somewhat homomorphic encryption would be useful *per se*, as a side contribution of our analysis we were able to entirely break it, undermining all of its security goals. These results have also been published [Barguil and Barreto 2015], and are summarized as an Appendix in the full thesis.

Finally, the complete thesis [Barguil 2015] is available for download from the archives of the University of São Paulo: `http://www.teses.usp.br/teses/disponiveis/3/3141/tde-12072016-083255/pt-br.php`.

References

Barguil, J. M. M. (2015). Efficient methods for lattice-based cryptography. Master's thesis, University of São Paulo.

Barguil, J. M. M. and Barreto, P. S. L. M. (2015). Security issues in Sarkar's e-cash protocol. *Information Processing Letters*, 115(11):801–803.

Barguil, J. M. M., Lino, R. Y., and Barreto, P. S. L. M. (2014). Efficient variants of the GGH-YK-M cryptosystem. In *Brazilian Symposium on Computer System and Information Security – SBSeg 2014*, Belo Horizonte, Brazil. SBC.

Bernstein, D. J. (2014). A subfield-logarithm attack against ideal lattices. Blog entry.

Campbell, P., Groves, M., and Shepherd, D. (2014). Soliloquy: A cautionary tale. In *ETSI 2nd Quantum-Safe Crypto Workshop*, Ottawa, Canada. ETSI.

Chenal, M. and Tang, Q. (2014). On key recovery attacks against existing somewhat homomorphic encryption schemes. In *International Conference on Cryptology and Information Security in Latin America – Latincrypt 2014*, Lecture Notes in Computer Science, Florianópolis, Brazil. Springer.

Cohen, H. (1993). *A course in computational algebraic number theory*, volume 138. Springer, Berlin, Germany.

de Barros, C. F. and Schechter, L. M. (2014). GGH may not be dead after all. In *XXXV Congresso Nacional de Matemática Aplicada e Computacional – CNMAC 2014*, Natal, Brazil. Sociedade Brasileira de Matemática Aplicada e Computacional – SBMAC.

Gentry, C. (2009a). *A fully homomorphic encryption scheme*. PhD thesis, Stanford University.

Gentry, C. (2009b). Fully homomorphic encryption using ideal lattices. In *XLI Annual ACM Symposium on Theory of Computing – STOC 2009*, volume 9, pages 169–178, Bethesda, MD, USA. ACM.

Gentry, C. and Halevi, S. (2011). Implementing Gentry's fully-homomorphic encryption scheme. In *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148, Tallinn, Estonia. Springer.

Goldreich, O., Goldwasser, S., and Halevi, S. (1997). Public-key cryptosystems from lattice reduction problems. In *Advances in Cryptology – CRYPTO '97*, pages 112–131. Springer, Santa Barbara, USA.

Loftus, J., May, A., Smart, N. P., and Vercauteren, F. (2012). On CCA-secure somewhat homomorphic encryption. In *International Conference on Selected Areas in Cryptography – SAC 2011*, volume 7118 of *Lecture Notes in Computer Science*, pages 55–72, Toronto, Canada. Springer.

Nguyen, P. (1999). Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from CRYPTO '97. In *Advances in Cryptology – CRYPTO '99*, pages 288–304, Santa Barbara, USA. Springer.

Sarkar, P. (2013). Multiple-use transferable e-cash. *International Journal of Computer Applications*, 77(6):35–38.

Smart, N. P. and Vercauteren, F. (2010). Fully homomorphic encryption with relatively small key and ciphertext sizes. In *XIII International Conference on Practice and Theory in Public Key Cryptography – PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443, Paris, France. Springer.

Szydlo, M. (2003). Hypercubic lattice reduction and analysis of GGH and NTRU signatures. In *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 433–448. Springer, Warsaw, Poland.

Yoshino, M. and Kunihiro, N. (2012). Improving GGH cryptosystem for large error vector. In *XI International Symposium on Information Theory and its Applications – ISITA 2012*, pages 416–420, Honolulu, HI, USA. IEEE.

