Analysis of One Fully Homomorphic Encryption Scheme in Client-Server Computing Scenario

Yang Li and Lihua Liu
(Corresponding author: Yang Li)

Department of Mathematics, Shanghai Maritime University,
Haigang Ave 1550, Shanghai, 201306, China
(Email: liyangsmu@126.com, liulh@shmtu.edu.cn)
(Received May 20, 2019; Revised and Accepted Dec. 6, 2019; First Online Feb. 1, 2020)

Abstract

Fully homomorphic encryption (FHE) is a useful primitive which allows anyone to perform arbitrary operations on encrypted data without decryption key, but any arithmetic (such as addition and multiplication) must be constrained to the underlying domain (finite fields or rings). In this paper, we revisit Dasgupta-Pal FHE scheme, and discuss its applications in client-server scenario. We find its calculation process cannot be practically completed due to the flawed relationship between the key size and the length of the plaintext. We would like to stress that the main purpose of cryptography using modular arithmetic is to obscure and dissipate redundancies in plaintext, not to perform numerical calculations. We think FHE could be of little significance in client-server scenario.

Keywords: Client-Server Computing; Fully Homomorphic Encryption; Polynomial Rings; Symmetric Encryption

1 Introduction

In recent years, fully homomorphic encryption (FHE) has flourished in various fields such as cloud computing, economic, searchable encryption, spam filtering, watermarking, e-voting, and medical services [16–18, 22, 29], which can be used to protect user privacy. Homomorphic encryption (HE) supporting either addition or multiplication (but not both), has some limitations for various scenarios.

In 1978, Rivest *et al.* [25] investigated the problem of privacy homomorphisms. Paillier's encryption [21] was a popular quasi-homomorphic encryption. In 2009, Gentry [10] presented a fully homomorphic encryption over ideal lattices which can allow anyone to evaluate the circuit on the encrypted data without the decryption key. But it is very slow because its key size is too large. In 2010, Gentry *et al.* [13] proposed a simple BGN-type cryptosystem from LWE. Shortly afterwards, Gentry and Halevi [11] presented a fully homomorphic encryption free of using depth-3 arithmetic circuits. In 2010, Dijk *et al.* [9] constructed an integer-based FHE scheme using modular operations. At Crypto'11, Coron *et al.* [7] provided a FHE scheme over integers with shorter public key. In 2013, Gupta and Sharma [14] presented a FHE scheme using a symmetric key with a smaller size. The works [3, 12, 28] tried to optimize ciphertext length and to improve efficiencies of some FHE schemes. In 2015, Nuida and Kurosawa constructed a fully homomorphic encryption scheme over integers with the message space $\mathbb{Z}_Q$ for any prime Q. The later works [2, 5, 6, 20, 27] studied the possible applications and optimizations of FHE. In 2019, Salavi *et al.* [26] considered the combinations of some traditional and modern cryptographic techniques for designing FHE schemes.

Cloud computing is a promising innovation for storing large amount of data. Sensitive data stored on cloud platforms are vulnerable to attacks by hackers and unauthorized parties. The works [19, 23, 24] discussed how to protect the security of cloud computing. In 2016, Jeng *et al.* [15] proposed a method to resist attacks against cloud storage services. Dasgupta and Pal [8] designed a symmetric FHE scheme based on polynomial rings. In 2018, Aganya and Sharma [1] improved the FHE scheme. Cao *et al.* [4] pointed out that some typical FHE schemes were not suitable for client-server or cloud computing scenarios.

In this paper, we revisit the Dasgupta-Pal FHE scheme and analyze its applications in client-server scenario. We would like to point out that the scheme fails to draw a clear line between numerical operations and modular operations. The relationship between the key size and the length of the plaintext is also confused. We want to stress that any computations operated on encrypted data should be constrained to the underlying domain. Otherwise, the related computations will generate some wrong outputs.

2 Dasgupta-Pal FHE Scheme

2.1 Description

The scheme can be described as follows.

KeyGen. For a security parameter l, generate secret key S_k. Pick a prime number of l bits and an even integer z of length γ. Set $R_k = z \cdot S_k$.

Enc. Pick n-degree polynomial $y(x), d(x)$ such that $m_p(x) \equiv y(x) \bmod S_k$, coefficients of $d(x)$ are integers of length l^a. Compute $c(x) = y(x) + S_k \cdot d(x)$.

Dec. Compute $c(x) \bmod S_k \bmod 2 = m_p(x)$.

Refresh. Compute $c'(x) = c(x) \bmod R_k$.

Clearly, we have

$$c_1(x) + c_2(x) \bmod S_k \bmod 2 = m_{p1}(x) + m_{p2}(x),$$
$$c_1(x) \cdot c_2(x) \bmod S_k \bmod 2 = m_{p1}(x) \cdot m_{p2}(x).$$

2.2 An Example in Client-Server Scenario

Suppose that $S_k = 13$. There are two numbers $m_1 = 69$ and $m_2 = 57$, a client asks a server to check if $(c_1 + c_2)(x) \bmod S_k \bmod 2$ is equal to $(m_{p1} + m_{p2})(x)$. Now, the scheme will encrypt m_1 and m_2 as follows.

$$\begin{aligned} m_1 &= 69 \to (1000101)_2 \to x^6 + x^2 + 1, y_1(x) \\ &\equiv m_1(x) \bmod 13 = 27x^6 + 14x^2 + 14. \end{aligned}$$

If pick $d(x) = 2650x^6 + 995x^5 + 259x^2 + 100$. Then

$$\begin{aligned} c_1(x) &= y_1(x) + S_k \cdot d_1(x) \\ &= 34477x^6 + 12935x^5 + 3381x^2 + 1314. \end{aligned}$$

$$\begin{aligned} m_{p1}(x) &= c_1(x) \bmod S_k \bmod 2 \\ &= (34477x^6 + 12935x^5 \\ &\quad +3381x^2 + 1314) \bmod 13 \bmod 2 \\ &= x^6 + x^2 + 1. \end{aligned}$$

$$\begin{aligned} m_2 &= 57 \to (111001)_2 \\ &\to x^5 + x^4 + x^3 + 1 \in R[x], \\ y_2(x) &\equiv m_2(x) \bmod 13 = 14x^5 + 27x^4 + 40x^3 + 14. \end{aligned}$$

If pick $d(x)$ $d_2(x) = 119x^5 + 224x^4 + 17x^3 + 2249x^2 + 36$. Then

$$\begin{aligned} c_2(x) &= y_2(x) + S_k \cdot d(x) \\ &= 1561x^5 + 2939x^4 + 261x^3 + 29237x^2 + 482. \end{aligned}$$

$$\begin{aligned} m_{p2}(x) &= c_2(x) \bmod S_k \bmod 2 \\ &= (1561x^5 + 2939x^4 + 261x^3 \\ &\quad +29237x^2 + 482) \bmod 13 \bmod 2 \\ &= x^5 + x^4 + x^3 + 1. \end{aligned}$$

2.3 Analysis

Suppose a, b are in the domain of one FHE encryption algorithm $E(\cdot)$, and $D(\cdot)$ is the corresponding decryption algorithm. Hence,

$$D(E(a) + E(b)) = D(E(a+b)) = (a+b) \bmod p$$
$$D(E(a) \cdot E(b)) = D(E(a \cdot b)) = (a \cdot b) \bmod p.$$

where p is the associated modular. Generally,

$$a + b \neq (a+b) \bmod p, \quad a \cdot b \neq (a \cdot b) \bmod p$$
$$a < b \not\Longrightarrow E(a) < E(b),$$
$$a \bmod q \bmod p \not\Longrightarrow a \bmod p \bmod q.$$

Any modular computations are constrained to the underlying domain, such as finite fields and rings. The above scheme uses only elementary modular operations on polynomial rings. But as we see that modular arithmetic is mainly used to obscure and dissipate redundancies in plaintext, not for common numerical calculations.

In the above example, the server who is not knowing the secret key will generate the following results for addition and multiplication.

$$\begin{aligned} & c_1(x) + c_2(x) \bmod 13 \bmod 2 \\ = \; & 34477x^6 + 14496x^5 + 2939x^4 + 261x^3 \\ & +32618x^2 + 1796 \bmod 13 \bmod 2 \\ = \; & x^6 + x^5 + x^4 + x^3 + x^2 \to (1111100)_2 = 124. \end{aligned}$$

But $m_1 + m_2 = 69 + 57 = 126 \neq 124$.

$$\begin{aligned} & c_1(x) \cdot c_2(x) \bmod S_k \bmod 2 \\ = \; & 53818597x^{11} + 121519438x^{10} + 47014462x^9 \\ & +1011380084x^8 + 383458336x^7 + 9936759x^6 \\ & +9168265x^5 + 98850297x^4 + 342954x^3 + 40047060x^2 \\ & +633348 \bmod 13 \bmod 2 \\ = \; & x^{11} + x^{10} + x^9 + x^7 + x^4 \\ & +x^3 + x^2 + 1 \to (111010011101)_2 \\ = \; & 3741. \end{aligned}$$

But But $m_1 \times m_2 = 69 \times 57 = 3933 \neq 3741$.

That is to say, the server will generate wrong outputs even for the two simple operations. In short, the scheme is not suitable for client-server scenario, because the overflow errors. Besides, it can not deal with any rounding errors in common numerical computations.

3 Aganya-Sharma FHE Scheme

In 2018, Aganya and Sharma [1] tried to improve Dasgupta-Pal FHE scheme. For readers' convenience, we now describe it as follows.

3.1 Description

The message space is $\{0,1\}^p$, where p is called batch size. l is a security parameter. Divide m into n substrings

$m^{(1)}, \ldots, m^{(n)}$, each of p bits. If the length of m is not an integer multiple of p, then add some padding bits 0s at its right end until it becomes a p bits string. For encryption purposes, each p bits string $m^{(i)}$ is converted into a decimal integer $m^{(i)}$.

KeyGen. Generate S_k, a prime number of l bits. Choose an even integer z of length γ, where $\gamma = \log_2 l$. Then set $R_k = z \cdot S_k$.

Enc. Choose a polynomial $y(x)$ of degree n such that $m_p(x) \equiv y(x) \bmod S_k$. Pick a polynomial $d(x)$ of degree n, with coefficients are integer of length l^a. Compute $c(x) = y(x) + S_k \cdot d(x)$.

Dec. Compute $c(x) \bmod S_k \bmod 2^p = m_p(x)$.

Refresh. Compute $c'(x) = c(x) \bmod R_k$.

3.2 Example

Suppose that $m_1 = 69$, $m_2 = 57$, $a = 5$. Set $S_k = 13$ be the secret key. $m_1 = 69 = (1000101)_2$, $m_2 = 57 = (111001)_2$. Pad $m^{(n)}$ with some 0s to ensure its length is an integer multiple of 3. Then $p = 3$ and the chunks are 001, 000, 101. In client-server scenario, its encrypting process is described as follows Table 1.

Table 1: m_1 and m_2 chunks

Encrypt 1:	$y_1^1 = 2453972,\ d_1^1 = 995,\ c_1^1 = 2466907$
Encrypt 0:	$y_1^2 = 445042,\ d_1^2 = 874,\ c_1^2 = 456404$
Encrypt 5:	$y_1^3 = 300708,\ d_1^3 = 742,\ c_1^3 = 310354$
Encrypt 0:	$y_2^1 = 445042,\ d_2^1 = 874,\ c_2^1 = 456404$
Encrypt 7:	$y_2^2 = 834750,\ d_2^2 = 731,\ c_2^2 = 844253$
Encrypt 1:	$y_2^3 = 2453972,\ d_2^3 = 995,\ c_2^3 = 2466907$

We shall obtain the coefficients 2923311, 1300657, 2777261. Its decrypting process will return the sum $(1111110)_2 = 126$.

3.3 Analysis

In the above example, the server who is not knowing the secret key will return wrong output for other texts. In fact, if choose $m_3 = 123 = (1111011)_2$ and $m_4 = 181 = (10110101)_2$. Adding 0s to the left ends of $m^{(3)}$ and $m^{(4)}$, we shall obtain the following two ciphertexts:

$$\underbrace{2466907, 844253, 315955}_{c_3} \quad \text{and} \quad \underbrace{458057, 453251, 310354}_{c_4}$$

For the function $f(x, y) = x + y$, the server evaluates it with the ciphertexts. It then returns values: 2924964, 1297504, 626309. Therefore, the decrypting process will return values:

$$\begin{aligned} 2924964 &\bmod S_k \bmod 2^p = 3, \\ 1297504 &\bmod S_k \bmod 2^p = 0, \\ 626309 &\bmod S_k \bmod 2^p = 0. \end{aligned}$$

Clearly, $(11000000)_2 = 192$, which is not the wanted value, 304. That is to say, the scheme has also confused the common numerical operations with the underlying modular operations. Namely, the improvement is also unsuitable for client-server scenario in cloud computing.

3.4 Further Discussions

What computations do you want to outsource privately? Backup your phone's contact directory to the cloud? Ask the cloud to solve a mathematic problem in your homework? Do a private web search? $\cdots$. It seems obvious that the daily computational tasks are rarely constrained to some prescribed modulus. Moreover, the client-server computing model can not deal with relational expressions which are defined over plain data, not over encrypted data. This is because

$$a < b \not\Longrightarrow E(a) < E(b), \quad E(a) < E(b) \not\Longrightarrow a < b.$$

In view of this drawback of FHE and the flaws discussed above, we think FHE seems inappropriate for the scenario of cloud computing.

4 Conclusion

In this note, we analyze two FHE schemes over polynomial rings in client-server scenario. We find the problem that what computations are worth delegating privately by individuals and companies to untrusted devices or servers remains untouched. We think the cloud computing community has not yet found a good for-profit model convincing individuals to pay for this or that computational service.

Acknowledgements

We thank the National Natural Science Foundation of China (Project 61411146001). The authors are very grateful to the reviewers for their valuable suggestions.

References

[1] K. Aganya and I. Sharma, "Symmetric fully homomorphic encryption scheme with polynomials operations," in *Proceedings of IEEE Second International Conference on Electronics, Communication and Aerospace Technology (ICECA'18)*, pp. 1954–1957, Mar. 2018.

[2] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.

[3] Z. Brakerskim, C. Gentry, and V. Vaikuntanathan, "(Leveled) Fully homomorphic encryption without bootstrapping," *ACM Transactions on Computation Theory*, vol. 6, no. 3, pp. 1–36, 2014.

[4] Z. J. Cao, L. H. Liu, and Y. Li, "Ruminations on fully homomorphic encryption in client-server computing scenario," *International Journal of Electronics and Information Engineering*, vol. 8, no. 1, pp. 32–39, 2018.

[5] G. Castagnos and F. Laguillaumie, "Linearly homomorphic encryption from DDH," in *Proceedings of Topics in Cryptology, The Cryptographer's Track at the RSA Conference (CT-RSA'15)*, pp. 487–505, Apr. 2015.

[6] J. H. Cheon and J. Kim, "A hybrid scheme of public-key encryption and somewhat homomorphic encryption," *IEEE Transaction on Information Forensics and Security*, vol. 10, no. 5, pp. 1052–1063, 2015.

[7] J. Coron and *et al.*, "Fully homomorphic encryption over the integers with shorter public keys," in *Proceedings of 31st Annual Cryptology Conference, Advances in Cryptology (CRYPTO'11)*, pp. 487–504, Aug. 2011.

[8] S. Dasgupta and S. K. Pal, "Design of a polynomial ring based symmetric homomorphic encryption scheme," *Perspectives in Science*, vol. 8, no. C, pp. 692–695, 2016.

[9] M. Dijk and *et al.*, "Fully homomorphic encryption over the integers," in *Proceedings of 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT'10)*, pp. 24–43, June 2010.

[10] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC'09)*, pp. 169–178, June 2009.

[11] C. Gentry and S. Halevi, "Fully homomorphic encryption without squashing using depth-3 arithmetic circuits," in *Proceedings of IEEE Annual Symposium on Foundations of Computer Science (FOCS'11)*, pp. 107–116, Palm Springs, Oct. 2011.

[12] C. Gentry and S. Halevi, "Implementing gentry's fully homomorphic encryption scheme," in *Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT'11)*, pp. 129–148, May 2011.

[13] C. Gentry, S. Halevi, and V. Vaikuntanathan, "A simple bgn-type cryptosystem from LWE," in *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT'10)*, pp. 506–522, May 2010.

[14] C. P. Gupta and I. Sharma, "A fully homomorphic encryption scheme with symmetric keys with application to private data processing in clouds," in *Proceedings of IEEE Fourth International Conference on the Network of the Future (NOF'13)*, pp. 23–25, Oct. 2013.

[15] F. G. Jeng and *et al.*, "On the security of privacy-preserving keyword searching for cloud storage services," *International Journal of Network Security*, vol. 18, no. 3, pp. 597–600, 2016.

[16] J. Kim, S. Kim, and J. H. Seo, "A new scale-invariant homomorphic encryption scheme," *Information Sciences*, vol. 422, pp. 177-187, 2017.

[17] C. L. Liu and C. W. Hsu, "Comment on 'improved secure RSA cryptosystem (ISRSAC) for data confidentiality in cloud'," *International Journal of Network Security*, vol. 21, no. 4, pp. 709–712, 2019.

[18] L. H. Liu and Z. J. Cao, "Analysis of two confidentiality-preserving image search schemes based on additive homomorphic encryption," *International Journal of Electronics and Information Engineering*, vol. 5, no. 1, pp. 1–5, 2016.

[19] L. H. Liu, Z. J. Cao, and C. Mao, "A note on one outsourcing scheme for big data access control in cloud," *International Journal of Electronics and Information Engineering*, vol. 9, no. 1, pp. 29–35, 2018.

[20] K. Nuida and K. Kurosawa, "(Batch) Fully homomorphic encryption over integers for non-binary message spaces," in *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT'15)*, pp. 537–555, Apr. 2015.

[21] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT'99)*, pp. 223–238, May 1999.

[22] M. M. Potey, C. A. Dhote, and D. H. Sharma, "Homomorphic encryption for security of cloud data," *Procedia Computer Science*, no. 79, pp. 175–181, 2016.

[23] V. S. Rao and N. Satyanarayana, "On multi-user based efficient computation outsourcing scheme and its application to cloud," *International Journal of Network Security*, vol. 21, no. 2, pp. 303–311, 2019.

[24] S. Rezaei, M. A. Doostari, and M. Bayat, "A lightweight and efficient data sharing scheme for cloud computing," *International Journal of Electronics and Information Engineering*, vol. 9, no. 2, pp. 115–131, 2018.

[25] R. Rivest, L. Adleman, and M. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of Secure Computation, Academia Press*, pp. 169–180, 1978. (`http://luca-giuzzi.unibs.it/corsi/Support/papers-cryptography/RAD78.pdf`)

[26] R. R. Salavi, M. M. Math, and U. P. Kulkarni, "A survey of various cryptographic techniques: From traditional cryptography to fully homomorphic encryption," in *Proceedings of Innovations in Computer Science and Engineering (ICICSE'19)*, pp. 295–305, Aug. 2019.

[27] C. Y. Tsai, C. Y. Liu, S. C. Tsaur, and M. S. Hwang, "A publicly verifiable authenticated encryption scheme based on factoring and discrete logarithms," *International Journal of Network Security*, vol. 19, no. 3, pp. 443–448, 2017.

[28] B. C. Wang, Y. Zhan, and Z. L. Zhang, "Cryptanalysis of a symmetric fully homomorphic encryption scheme," *IEEE Transaction on Information and Forensics Security*, vol. 13, pp. 1460–1467, 2018.

[29] C. H. Wei, M. S. Hwang, and A. Y. Chin, "A secure privacy and authentication protocol for passive rfid tags," *International Journal of Mobile Communications*, vol. 15, no. 3, pp. 266–277, 2017.

Biography

Yang Li is currently pursuing his M.S. degree from Department of Mathematics, Shanghai Maritime university. His research interests include combinatorics and cryptography.

Lihua Liu is an associate professor with Department of Mathematics at Shanghai Maritime University. She received her Ph.D. degree in applied mathematics from Shanghai Jiao Tong University. Her research interests include combinatorics, cryptography and information security.