Processing Encrypted Data

CAT 6427 Secret Sharing Protocols Spring 2017 – Assignment 3

Maciej Medyk

Computer Science Graduate Student

Florida Atlantic University

Boca Raton, United States

Abstract — **As many businesses are moving to cloud-based data stores to provide more convenience and accessibility for their customers, the data security and integrity had to be insured through data encryption. However, security came at very high computational cost since decryption of data had to be done prior to any processing of stored data. In recent years, computer scientists have worked on a concept of processing encrypted data that does not require prior decryption and would result in same outcome as if computation would be done on unencrypted data. The purpose of this paper is to evaluate how processing on encrypted data is done, and compare advantages and disadvantages between fully homomorphic encryption and partially homomorphic encryption.**

I. Introduction

When users save their data into the cloud storage, the data is often encrypted with various algorithms, but in order to process the data for running sets of algorithmic operations the data needs to be unencrypted before computation takes place. For some numerical data like banking transactions or military coordinates of assets, it would be practical to encrypt the data using homomorphic encryption schemes that allow for processing to happen on encrypted data without the need for prior decryption. To achieve this goal, application may use secure multi-party computation or homomorphic encryption which is currently divided into full homomorphic encryption (FHE) and partially homomorphic encryption (PHE).

II. Background

Fully homomorphic encryption (FHE) was introduced by Craig Gentry in year 2009 in his dissertation thesis and his encryption supports arbitrary amount of computation over encrypted data while remaining secure. To this point, all other encryption techniques were only able to support single computation at the time rather than chain of computations [1].

Purpose of Gentry scheme was to support both multiplication and addition at the same time by using Boolean algebra. Since 2009, there have been many improvements made to fully homomorphic encryption in variety of different algorithms like Fully Homomorphic Encryption in Honest but Curious Clouds [2] or Flexible Fully Homomorphic Encryption [3]; however, non of them achieve efficiency to be used in practical applications.

There are also have partially homomorphic encryption (PHE), which concept have been introduced by Ronald Rivest in 1978 [4]. There are multiple PHE cryptosystems that use different techniques to process encrypted data developed by Taher El Gamal, Paillier, Goldwasser and Micali [5]. We will now further discuss their advantages and disadvantages and compare them to full homomorphic algorithms.

III. Main Body

El Gamal cryptosystem was proposed in 1984 and it supports multiplication over encrypted data. Below is multiplication formula for $m_1 \times m_2$.

$$c_1 c_2 = (\alpha^{k_1}\alpha^{k_2} \bmod p, ((m_1 \cdot y^{k_1})(m_2 \cdot y^{k_2})) \bmod p)$$

$$= (\alpha^{k_1+k_2}, m_1 m_2 \cdot y^{k_1+k_2}) \bmod p$$

It is based on solving discreate logarithms. Its encryption is probabilistic meaning that unencrypted message can be encrypted to many different ciphers messages. The disadvantage to this encryption scheme is that size of the encrypted text was much larger than the size of plain text [6,7,8]. Practical use of El Gamal is in hybrid cryptosystems, where convenience of public key is coupled with efficiency of symmetric key cryptosystem. El Gamal is usually slower than other symmetric systems with similar level of security [9].

On the other hand, Goldwasser-Micali cryptosystem which was proposed in 1982 was first probabilistic encryption scheme that introduced the term of semantic security. Semantic security is achieved when cyphertext length of the message does not give any indication to the context of the message. The encryption supports addition operation.

Below is the addition formula for $x_1 + x_2$.

$$c_1 c_2 = (-1^{x_1} {r_1}^2)(-1^{x_2} {r_2}^2) \; mod \; 2$$

$$= -1^{(x_1+x_2)} (r_1 r_2)^2 \; mod \; 2$$

The scheme is based on quadratic residues complexity with respect to composite number n = pq where p and q were distinct prime numbers. The downside to this scheme is that it is not efficient cryptosystem and cypher message may be much larger than original raw text [8,10].

Lastly, Paillier cryptosystem was proposed in 1999 and it is based on decisional composite residuality class. It supports addition, subtraction, and multiplication by a constant. Below is the addition formula for $m_1 + m_2$

$$c_1 c_2 \; mod \; n^2 = g^{m_1} r_1^n g^{m_2} r_2^n \;\; mod \; n^2$$

$$= g^{m_1+m_2} r_1^n r_2^n \;\; mod \; n^2$$

Paillier cryptosystem has found its use in electronic voting and electronic monetary transactions, where semantic security is one of the considerations [8,11].

Fully homomorphic encryption introduced by Craig Gentry using lattice-based cryptography was a breakthrough in cryptography as data was able to be maintained through multiple amounts of operations. Any operation uses Boolean circuit where message are set into sequence of binary digits. To convert simple application into Boolean circuit is computationally very expensive and is limited to evaluating low-degree polynomials over encrypted data. Gentry later modified his scheme to make it able to evaluate its own decryption circuit and by refreshing ciphertext he was able to achieve continuous ability to compute addition and multiplication operations. This process can also be used for a search for portion of the text inside larger portion of text; however, the process can be tedious and very time consuming as process is comparing searched text bit by bit [6,8].

A newer adaptation of fully homomorphic encryption was introduced by Dongxi Liu in 2016 research is a scheme that is designed for cloud environments that expect honest but curious participants rather than malicious ones. There are four steps in FHE scheme: key generation, encryption, decryption, and homomorphic operations. He achieves a better performance by eliminating security against brute attacks. In this scheme, the cipher text is converted to a vector allowing vector additions and scalar multiplications which are more efficient to implement and calculate. The drawback to this approach is lack of security against brute force attacks [2].

Lastly, a batch fully homomorphic encryption scheme proposed by Zvika Brakerski in 2012 which is based the scheme over the integers with the same scale-invariant. The scheme contains a single modulus whose size is linear, rather than exponential, in depth of the evaluated. This scheme then can be transformed into pure FHE using bootstrapping and its security is based on Greatest Common Denominator problem [3,12].

IV. Conclusion

Processing of encrypted data only recently became a hot topic due to the rise in cloud environment; however, there are still many issues with efficiency of encryption. As Fully Homomorphic Encryption offers ability to do many operations on cyphertext, it is highly inefficient. There are some variances of FHE that do try to address inefficiencies, but sacrifice many security aspects. Therefore, FHE has not been adopted yet for any practical use due to its disadvantages. On the other hand, Partially Homomorphic Encryption is efficient, but has limitations in the amount of computations it offers on ciphertext. Many of PHE schemas are already being used in electronic voting or in monetary transactions. Both FHE and PHE methods offer secure techniques of conducting arithmetic operations on encrypted data without need of decryption of data prior to computation. I believe that with time we will be able to improve FHE to be a viable option, but from the two homomorphic encryption schema types only one that is being used is Partially Homomorphic Encryption and from the three reviewed here seems like Paillier is the one offering the most options while providing security, justifying its use in current applications.

References

[1] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. Paper presented at the 169-178. doi:10.1145/1536414.1536440

[2] Liu, D. (2016). Efficient processing of encrypted data in honest-but-curious clouds. Paper presented at the 970-974. doi:10.1109/CLOUD.2016.0147

[3] Ma, C., Li, J., & Du, G. (2016). A flexible fully homomorphic encryption. Wireless Personal Communications, doi:10.1007/s11277-016-3796-5

[4] R. Rivest, L. Adleman, and M. Dertouzos, On data banks and privacy homomorphisms, In: Foundations of Secure Computation, New York: Academic Press, pp.169–179, 1978

[5] Islam, N., Puech, W., & Brouzet, R. (2011). Comparison of three solutions to correct erroneous blocks to extract an image of a multiplicative homomorphic cryptosystem. Paper presented at the , 7880(1) doi:10.1117/12.873319

[6] Wikipedia, https://en.wikipedia.org/wiki/ homomorphic_encryption

[7] Wikipedia, https://en.wikipedia.org/wiki/ ElGamal_encryption

[8] Saleh, E., Alsa'deh, A., Kayed, A., & Meinel, C. (2016). Processing over encrypted data: Between theory and practice. ACM SIGMOD Record, 45(3), 5-16. doi:10.1145/3022860.3022862

[9] R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multiauthority election scheme. In EUROCRYPT, pages 103–118, NY, USA, 1997.

[10] Wikipedia, https://en.wikipedia.org/wiki/Goldwasser-Micali_cryptosystem

[11] Wikipedia, https://en.wikipedia.org/wiki/Paillier_ Cryptosystem

[12] Cheon, J., Coron, J., Kim, J., Lee, M., Lepoint, T., Tibouchi, M., & Yun, A. (2013). Batch fully homomorphic encryption over the integers. Paper presented at the 315-335.