On Enhancing Security for Division Homomorphism with ElGamal

Marius Silaghi
Florida Tech University
Melbourne, FL, USA
msilaghi@fit.edu

Ameerah Alsulami
Florida Tech University
Melbourne, FL, USA
aalsulami2019@my.fit.edu

Abstract

Secure auctions and machine learning in cloud increasingly employs multi-party and homomorphic encryption support. A modification to Elgamal public key cryptosystem was shown to enable homomorphic division using an encoding of plaintext as fractions with numerator and denominator encrypted separately. However we notice that unlike for other homomorphic cryptography schemes, the obtained division homomorphism allows for the retrieval of the input secrets from the result of the division. Since this cancels the benefit of the encryption, we propose the introduction of a masking operation based on random factors and discuss its success with operations in $\mathbb{Z}_p$ and $\mathbb{Q}$.

Introduction

The use of homomorphic encryption in machine learning on the cloud enhances privacy, decreases risk of data breaches, and increases confidence in machine learning models (Pulido-Gaytan et al. 2021). The ElGamal encryption scheme is a type of homomorphic encryption. However, the security of the Division Homomorphism of the ElGamal encryption scheme proposed in (Sidorov, Wei, and Ng 2022) needs to be addressed.

Background

Homomorphic schemes can solve practical issues in cryptography while keeping the data encrypted (Alaya, Laouamer, and Msilini 2020). Therefore, the development of homomorphic encryption has opened new avenues for secure computation on encrypted data. The ElGamal public key encryption scheme, a popular form of homomorphic encryption, operates on the principles of modular exponentiation (Tran et al. 2020). The security of the recently proposed (Alaya, Laouamer, and Msilini 2020) division homomorphism in the ElGamal encryption scheme is a topic of concern.

Back to the past, the earliest encryption techniques date back to ancient civilizations, such as the Spartans and the Greeks (Mollin 2005). One of the most famous historical encryption techniques is the Caesar cipher, which was used by Julius Caesar to encode messages (Dewangga, Purboyo, and Nugrahaeni 2017). In this cipher, each letter in the message is shifted a fixed number of positions down the alphabet. Since then, numerous cryptographic methods have been developed, including symmetric and public encryption (William et al. 2022). Symmetric-key encryption employs a single secret key for both encryption and decryption. This necessitates securely sharing the key between involved parties, a practical challenge (Chandramouli, Iorga, and Chokhani 2014). On the other hand, public-key encryption uses two keys - a public key for encryption and a private key for decryption. The public key can be freely distributed, while the private key must be kept secret. This allows for secure communication without the need for key exchange (Khalifa et al. 2004). Despite their usefulness, traditional encryption methods have limitations when it comes to processing encrypted data. Because before performing any computation on encrypted data, it requires decrypting the data first, then performing the computation, and finally reencrypt the data. This exposes the data to potential security risks and can be computationally expensive. This is where homomorphic encryption comes in.

The classification of homomorphic encryption methods is based on the number and type of mathematical operations that can be carried out on encrypted data. There are three levels of homomorphic encryption, with Partially Homomorphic Encryption (PHE) being the first, which only allows one type of operation infinite times on the encrypted data, such as addition or multiplication, but not both. The second level is Somewhat Homomorphic Encryption (SHE), which permits multiple operations to be performed on the encrypted data but has limitations, such as on the number of some operations. As a result, the types and number of operations that can be performed are restricted. The final level is Fully Homomorphic Encryption (FHE), which allows unlimited operations to be performed on encrypted data but is the most computationally expensive and may not be practical for all use cases (Fawaz et al. 2021; Sidorov, Wei, and Ng 2022).

There are two types of Partially Homomorphic Encryption (PHE). First, Additive Homomorphic Encryption (ex, Paillier, BGV cryptosystem) has the ability to sum encrypted numerical values while keeping the original numbers confidential. Second, Multiplicative Homomorphic Encryption which can multiply encrypted values without revealing the

original values (Gao et al. 2020). The RSA and El-Gamal cryptographic systems are examples of schemes that possess the property of multiplicative homomorphism (Alkharji, Liu, and Washington 2016).

Paillier cryptosystem is an example of Additive Homomorphic Encryption scheme introduced in 1999 by Pascal Paillier. The Paillier cryptosystem is based on the computational difficulty of the decisional composite residuosity assumption, which is a variant of the decisional Diffie-Hellman assumption (Yaji, Bangera, and Neelima 2018). In MPC protocols using Shamir polynomial shares, the data is divided into shares using a secret sharing scheme based on Shamir polynomials. Each share is then encrypted using Paillier encryption (Demmler, Schneider, and Zohner 2015). The encrypted shares are sent to the other parties, who can perform computations on them. The final result is obtained by combining the results of the computations on the encrypted shares and then applying the inverse of the secret sharing scheme.

MPC protocols have many applications, such as secure data processing, privacy-preserving machine learning, and secure cloud computing (Mohassel and Zhang 2017). However, HE and MPC are computationally intensive, and the size of ciphertexts can be significantly larger than plaintexts, which can lead to performance and storage challenges.

Indeed, it should be recognized that while PHE has its benefits, it also has certain restrictions, such as its inability to handle certain types of computations. Nonetheless, PHE can prove valuable when maintaining data confidentiality is essential, such as in secure cloud computing or privacy-protecting data analysis (Acar et al. 2018).

The current state of the field highlights the need for further research on the enhancement of security for the Division Homomorphism with the ElGamal encryption scheme.

Analysis

With a public key p, g, y, the Elgamal Encryption of a message m is given by: $E(m) = (m * y^x \bmod p, g^x \bmod p) = (\alpha, \beta)$ where x is a value chosen randomly at each encryption, and where

$$\alpha = m * y^x \bmod p$$

and

$$\beta = g^x \bmod p$$

This scheme is partially homomorphic. It offers, based on

$$E(m_1) = (m_1 * y^{x_1} \bmod p, g^{x_1} \bmod p) = (\alpha_1, \beta_1)$$

and

$$E(m_2) = (m_2 * y^{x_2} \bmod p, g^{x_2} \bmod p) = (\alpha_2, \beta_2),$$

the result:

$$E(m_1 * m_2) = (\alpha_1\alpha_2, \beta_1\beta_2) = E(m_1) \cdot E(m_2)$$

A new version scheme E' for Elgamal was proposed in (Sidorov, Wei, and Ng 2022) for division with ElGamal encryption by encoding values as fractions $\frac{a*10^k}{10^k}$.

For E' one selects an integer k and one defined

$$E'(m) = E'\left(\frac{m * 10^k}{10^k}\right) = (E(m_1), E(m_2)).$$

where $m_1 = m * 10^k$ and $m_2 = 10^k$.

At decryption, the two components of the result, m_1 and m_2, are decrypted separately and $m = \frac{m_1}{m_2}$. As such now we have $E'(a) = E'\left(\frac{a*10^k}{10^k}\right) = (E(a_1), E(a_2))$, while $E'(b) = E'\left(\frac{b*10^k}{10^k}\right) = (E(b_1), E(b_2))$.

The product is now:

$$E'(a * b) = (E(a_1 * b_1), E(a_2 * b_2))$$

In (Sidorov, Wei, and Ng 2022) it was proposed to also compute division as:

$$E'(a/b) = (E(a_1 * b_2), E(a_2 * b_1))$$

Unfortunately, this approach no longer guarantees that from $\mathrm{E}'(\mathrm{a} * \mathrm{b})$, at decryption, one would no longer be able to retrieve the original values a and b.

For example, if $a = 5$ and $b = 6$ while E' uses $k = 3$, then $E'(a/b)$ is decrypted by independently obtaining the numerator 5000000 and denominator 6000000, leading to the result $\frac{5000000}{6000000}$.

However, from this decryption process it is obvious that the inputs were $a = 5$ and $b = 6$.

An additional blinding step is required in order to hide the input secrets when the result is decrypted. An additional radom non-zero fraction $\frac{c}{c}$ has to be generated and $(E(c), E(c))$ must be multiplied into the mix by the party performing the multiplication.

The result will be:

$$E'(a/b) = (E(a_1 * b_2 * c), E(a_2 * b_1 * c))$$

If the operation of division is intended to be performed in $\mathbb{Z}_p$, then the value of c can be any non-zero random value in $\mathbb{Z}_p$.

However, if the computation is intended to be performed in $\mathbb{Q}$, then c has to be selected carefully to avoid overflows and to contain many prime factors that are likely to occur in a and b.

Conclusion

In conclusion, the modification to the Elgamal public key cryptosystem proposed in this research paper has successfully enabled homomorphic division. However, it has been noted that the division homomorphism is invertible, which negates the security benefits of encryption. To address this issue, we proposed the introduction of a masking operation based on random factors and demonstrated its effectiveness in operations over both $\mathbb{Z}_p$ and $\mathbb{Q}$. Our research highlights the need for further improvement in homomorphic cryptography to ensure secure and efficient computations on encrypted data.

References

Acar, A.; Aksu, H.; Uluagac, A. S.; and Conti, M. 2018. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)* 51(4):1–35.

Alaya, B.; Laouamer, L.; and Msilini, N. 2020. Homomorphic encryption systems statement: Trends and challenges. *Computer Science Review* 36:100235.

Alkharji, M.; Liu, H.; and Washington, C. 2016. Homomorphic encryption algorithms and schemes for secure computations in the cloud. In *Proceedings of 2016 International Conference on Secure Computing and Technology*, 19.

Chandramouli, R.; Iorga, M.; and Chokhani, S. 2014. *Cryptographic key management issues and challenges in cloud services*. Springer.

Demmler, D.; Schneider, T.; and Zohner, M. 2015. Aby-a framework for efficient mixed-protocol secure two-party computation. In *NDSS*.

Dewangga, I.; Purboyo, T. W.; and Nugrahaeni, R. A. 2017. A new approach of data hiding in bmp image using lsb steganography and caesar vigenere cipher cryptography. *International Journal of Applied Engineering Research* 12(21):10626–10636.

Fawaz, S. M.; Belal, N.; ElRefaey, A.; and Fakhr, M. W. 2021. A comparative study of homomorphic encryption schemes using microsoft seal. In *Journal of Physics: Conference Series*, volume 2128, 012021. IOP Publishing.

Gao, C.-z.; Li, J.; Xia, S.; Choo, K.-K. R.; Lou, W.; and Dong, C. 2020. Mas-encryption and its applications in privacy-preserving classifiers. *IEEE Transactions on Knowledge and Data Engineering* 34(5):2306–2323.

Khalifa, O. O.; Islam, M. R.; Khan, S.; and Shebani, M. S. 2004. Communications cryptography. In *2004 RF and Microwave Conference (IEEE Cat. No. 04EX924)*, 220–223. IEEE.

Mohassel, P., and Zhang, Y. 2017. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE symposium on security and privacy (SP)*, 19–38. IEEE.

Mollin, R. A. 2005. *Codes: the guide to secrecy from ancient to modern times*. Chapman and Hall/CRC.

Pulido-Gaytan, L. B.; Tchernykh, A.; Cortés-Mendoza, J. M.; Babenko, M.; and Radchenko, G. 2021. A survey on privacy-preserving machine learning with fully homomorphic encryption. In *High Performance Computing: 7th Latin American Conference, CARLA 2020, Cuenca, Ecuador, September 2–4, 2020, Revised Selected Papers 7*, 115–129. Springer.

Sidorov, V.; Wei, E. Y. F.; and Ng, W. K. 2022. Comprehensive performance analysis of homomorphic cryptosystems for practical data processing. *arXiv preprint arXiv:2202.02960*.

Tran, J.; Farokhi, F.; Cantoni, M.; and Shames, I. 2020. Implementing homomorphic encryption based secure feedback control. *Control Engineering Practice* 97:104350.

William, P.; Choubey, A.; Chhabra, G.; Bhattacharya, R.; Vengatesan, K.; and Choubey, S. 2022. Assessment of hybrid cryptographic algorithm for secure sharing of textual and pictorial content. In *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, 918–922. IEEE.

Yaji, S.; Bangera, K.; and Neelima, B. 2018. Privacy preserving in blockchain based on partial homomorphic encryption system for ai applications. In *2018 IEEE 25th International Conference on High Performance Computing Workshops (HiPCW)*, 81–85. IEEE.