Homomorphic Encryption

Sean Brieffies
CSU34031 Research Paper
Student Number 16339746
Exam Number 44301
brieffis@tcd.ie

***Abstract*—Homomorphic Encrytion has existed in some form for a number of decades in popular encryption schemes such as RSA and ElGamal. While these schemes provided some homomorphic functionality, a fully homomorphic encryption (FHE) scheme eluded researchers until Craig Gentry's 2009 Doctural Dissertation. In this he described a scheme which made use of a pre-existing type of homomorphic encryption known as somewhat homomorphic encryption. He paired this with a new 'bootstrapping' technique to achieve the first ever fully homomorphic encryption scheme. Since then, research in this area has exploded as FHE promises to enable data to remain encrypted during its entire lifecycle online. The nature of most modern FHE schemes are also secure against quantum computers which makes this area of cryptography of great interest to those looking to secure our present and future. A standardisation movement headed by Microsoft, IBM and Duality Technologies took form in 2017 and is continuing to pursue co-ordinated efforts to build on the gains the homomorphic encryption has made since Gentry's seminal work was published just over a decade ago. Here I review past and current developments and lay out a number of practical scenarios which would benefit from a FHE scheme. It was concluded that homomorphic encryption has strong potential to secure data online and future-proof the security of our data online.**

***Index Terms*—SWHE, Generation, Practical, RLWE, Homomorphicencryption.org**

I. Introduction

Homomorphic (same form) Encryption is a form of encryption which allows for computations to be performed directly on ciphertexts without first needing to decrypt that text. The result remains encrypted and when later decrypted, is the same as if the computation had been performed on plaintext. It achieves this by preserving the underlying mathematical structure of the plaintext.

A simple example of this would be encrypting numbers 200 as 2 and 100 as 1. If we divide these numbers, we get a result of 2, which is the same as if the calculation had been performed on the original plaintext numbers.

Unfortunately, practical homomorphic encryption schemes are not this simple to achieve. There is somewhat of a contradiction in homomorphic encryption: it requires enough structure in the ciphertexts to carry out computations but also enough randomness that an attacker cannot bypass the encryption. The idea of homomorphic encryption was first proposed in 1978 by Rivest, Adleman and Dertozous [1], just one year after Rivest, Adleman and Shamir invented the still-widely-used RSA cryptosystem. But it was not until thirty-one years later with Craig Gentry's 2009 doctoral dissertation that the first fully homomorphic encryption (FHE) scheme was realised [2]. While this marked a huge advance in homomorphic encryption, it was not even remotely fast enough to be implemented in a real-life environment. Gentry estimated that a Google search encrypted with his scheme would take roughly a trillion times longer, or 31,688 years for a one second search [3].

The attraction towards homomorphic encryption lies in the fact that it allows for data to be stored and manipulated by untrusted parties without revealing anything about the data. For example, an individual or organisation could encrypt information using a private key, store the encrypted information on the cloud, manipulate the information using cloud infrastructure (which could be far superior to their own local hardware), download the encrypted results and then decrypt them locally. During this whole process the untrusted party gains no access to the plaintext. The cloud service could potentially view and manipulate the ciphertexts but without the private encryption key they cannot decrypt it to learn anything meaningful.

In the decade since Gentry's initial leap, new advances have brought multiple orders of magnitude performance gains to FHE, but it still lags behind traditional encryption protocols in terms of performance and efficiency. There is, however, a clear desire to continue this forward momentum as FHE has huge potential to increase security and data privacy. This is especially important given the continued growth of cloud computing, big data both of which are directly at odds with a growing desire for more personal privacy online.

II. Types of Homomorphic Encryption

There are three forms of homomorphic encryption:

A. Partially Homomorphic Encryption (PHE)

Only allows for one of either addition or multiplication to be performed on encrypted data. The RSA (unpadded) and ElGamal cryptography systems are homomorphic under multiplication. PHE could be used in more mathematically trivial situations such as a voting system, where only addition is required [4]. But in more complex areas such as finance or medicine a combination of addition and multiplication is usually required so PHE is not suitable.

B. Somewhat Homomorphic Encryption (SWHE)

Allows for both addition and multiplication to be performed on encrypted data, but only a limited number of operations can be performed. Noise is introduced with each calculation

and once this reaches a certain threshold, the encrypted value can no longer be decrypted. Additions add a small amount of noise, but multiplications cause the noise to multiply itself. It is for this reason that SWHE schemes typically support many additions but only a few multiplications. SWHE can be used for more simple computations such as calculating the mean or standard deviation of a data set as these can be achieved with zero and one multiplications respectively [5]. For more advanced calculations in areas such as finance, science and machine learning a more flexible solution is required, hence the desire for FHE, which has been called the 'holy grail of cryptography [6].

C. Fully Homomorphic Encryption (FHE)

Allows for an arbitrary number of additions and multiplications to be performed on encrypted data. Gentry built his FHE scheme by starting with a SWHE scheme and then using an additional 'bootstrapping' step to eliminate the noise generated by homomorphic calculations. It is inherently slower than SWHE due to the overhead of bootstrapping. FHE fixes the shortcomings of PHE and SWHE, making it a secure way to perform complex computations on large data sets online. This is why it is such a desirable encryption scheme and it will be the main focus of this paper.

D. Summary

TABLE I
TYPES OF HOMOMORPHIC ENCRYPTION

Type	***Operation***	***# Operations***	***Use Case***
PHE	Add/Mul	Limited	Secure Voting System
SWHE	Add+Mul	Limited	Simple Calculations
FHE	Add+Mul	Unlimited	Any Calculations

III. WHY IS FHE IN DEMAND?

As mentioned earlier, FHE allows for computations to be performed on ciphertext, there are a great many possible applications of this in today's world. In recent years we have seen an explosion of data generation, collection, and analytics. In 2012 IBM reported that 2.5 exabytes (quintillion bytes) were produced each day [7]. From 2010 – 2012 90% of the data in the world was created. Since then, smartphone sales have more than doubled and as of August 2018, there are over 17 billion connected devices. 7 billion of these are Internet of Things (IoT) devices, a market which is predicted to average growth of 28.86% every year until 2026. These devices are collecting more and more personal information on users. The World Economic Forum have predicted that we will generate 464 exabytes of data per day by 2025 [11]. This explosion of data has only heightened the fear of data breaches for users. For example, in 2013, 2 separate breaches saw 3 billion Yahoo accounts being compromised. Hackers gained access to names, phone numbers, security questions and even hash passwords. Fitness wearables collect live locations of users and sensitive health statistics. While these are encrypted and stored on the cloud, the security of these systems is out of the control of the user as everything is 'up in the cloud'. Increased public pressure for control of data has led to recent legislative changes like the General Data Protection Regulation (GDPR). What if homomorphic encryption could help to solve this conundrum? Theoretically, users could take advantage of FHE to ensure that only encrypted versions of their information live online which can only then be decrypted with their own personal private key. This would give users far more control over what information they are willing to share with the various internet services that they use; and may even increase the amount of information they are willing to share given that these services cannot view the plaintext of their data.

IV. HISTORY OF MODERN FHE

A. First Generation FHE

As previously mentioned, the possiblility of FHE was first theorised in 1978 but it was not until 2009 when the first FHE scheme was created [2]. Craig Gentry accomplished this by converting a SWHE scheme to a FHE scheme by introducing a 'bootstrapping' step. For a SWHE scheme to be bootstrappable, it must be able to evaluate its own decryption and then at least one more operation. What this step essentially does is to 'refresh' the ciphertext once the noise levels reach a certain threshold. One can then continue to operate on the ciphertext until the threshold is hit again, and we repeat the process as many times as necessary. This allows for an arbitrary number of additions and multiplications to be performed on the ciphertext.

B. Second Generation FHE

This began with work by Brakerski, Gentry and Vaikuntanathan (BGV) in 2011. Security based on the hardness of the "Ring Learning with Errors" problem [8]. They used a new method of noise reduction known as Modulus Switching which allows for exponentially more multiplications before the noise threshold is reached. This eliminates the need for Bootstrapping. Ironically, it is still performed but instead as an optimisation for larger circuits. Hence it still follows the basic blueprint of Gentry's original construction, namely first construct a SWHE scheme and convert it to FHE scheme using bootstrapping. As well as featuring lower noise growth during homomorphic computations, a new 'relinearization' step reduces the size of ciphertext from quadratic to linear.

C. Third Generation FHE

In 2013, Gentry, Sahai and Water (GSW) published a new FHE technique which avoids the need for the expensive 'relinearization' step, making it faster and conceptually simpler than the previous generation [9]. A Google search encrypted with this new scheme would take roughly 12 days. This would still be one million time longer than a standard search, but still represented an enormous improvement from just 4 years earlier when it took one trillion times longer.

V. Security and Efficiency of FHE Techniques

Most well-known cryptography techniques achieve their security by using the hardness of certain mathematical problems e.g. the RSA cryptosystem relies on the difficulty of factoring large prime numbers. Currently, there are four main bases for FHE: Ideal lattice based, integer based, ring learning with errors (RLWE) based and elliptic curve cryptography (ECC) based [10].

A. Ideal lattice

Gentry based the security of his original FHE scheme on problems with lattices [2]. Lattice based cryptographic systems allow for the decryption function to have a low enough complexity to facilitate the bootstrapping procedure. Lattices can be viewed as an infinitely big grid of regularly spaced points. Gentry's solution made use of ideal lattices, which are additively and multiplicatively homomorphic and so this was obviously of interest to Gentry when he was seeking to create a cryptographic scheme which also satisfied these same properties. The hardness of this scheme is based on the 'closet vector' and 'subset sum' problems. Ideal lattice based FHE is more secure and more efficient than Integer based but is behind RLWE and ECC based implementations in these areas.

B. Integer

Integer based FHE relies on the hardness of approximate GCD calculation. This is the most conceptually simple base from which we can construct an FHE scheme but comes with certain limitations. Namely, it is the least secure and least efficient of the four choices.

C. RLWE

FHE schemes based on RLWE are the most practical. These schemes are simpler than ideal lattice based schemes while also being more secure and efficient. The RLWE problem is considered to be secure against quantum computers, making it more secure than factorisation techniques such as RSA or discrete logarithm techniques such as ECC. NIST's 'post-quantum cryptography standardisation project' received numerous hard lattice problem submissions which are like the RLWE technique used in FHE, making RLWE based FHE an ideal candidate to ensure post-quantum security [11].

D. Elliptic curve cryptography

Elliptic curve cryptography based FHE schemes base their hardness on the Elliptic Curve Discrete Logarithm Problem. They are the most secure, efficient and are also simpler to implement compared to RLWE or ideal lattice based schemes. They do suffer from a key limitation in that they do not support floating point calculations. This limits their usability in almost all data-heavy fields and so a RLWE based approach is the current standard.

VI. Practical Uses of Homomorphic Encryption

A. Secure Data Outsourcing

Probably the most talked about benefit of FHE is that it allows for the storage and analysis of sensitive data to be securely outsourced. As Gentry said, it enables us "to delegate processing of your data without giving away access to it". One could locally encrypt their data using a FHE scheme, upload the result to the cloud, perform homomorphic computations on it, download it and then finally decrypt it. This would further empower the cloud and the user. Cloud computing today is, in many ways, like the use of 'main-frame' computers in the past, in that a powerful central computer runs workloads and interfaces with weaker local machines. There have been many recent developments such as video game streaming which share in this core principle. Cloud computing could allow organisations to run statistical analysis on more powerful machines, saving time and money. A current hurdle to taking advantage of this avenue is the fact that industries which collect sensitive information, such as the medical industry, are often bound by law to only store data online if it is encrypted. This means that if, for example, a hospital wishes to run computations on their data, they must first download it, decrypt it, and then run computations on their weaker local machines. A FHE scheme would be highly beneficial to this situation as medical institutions could save time, and therefore money, and also have less of a need to spend large amounts of money of expensive local hardware all the while staying inside the law and protecting the sensitive data of their patients.

B. Private Information Retrieval

Homomorphic Encryption could be very useful for Private Information Retrieval (PIR), meaning a protocol where a client can retrieve an item from a database without revealing which item has been retrieved [12]. It would allow for search to be performed on a database without first having to decrypt the data. The only overhead being that the client would have to first encrypt their search query. While this does ensure security on the cloud, it limits functionality. FHE would allow for searching and private retrieval of encrypted data from a database. This could be very useful in sensitive areas such as medicine or a public government database.

C. Data pooling

A FHE scheme also allows for users to pool their data without having to actually reveal the underlying the data. This could be very useful in areas such as machine learning. Machine learning often involves separate parties with very distinct roles, one or more parties who own the data and one who owns the learning model. FHE would allow the data-owning parties to share their valuable data for use on the learning model without revealing their data to any of the other parties. In December 2018 Intel released an open source package called 'HE Transformer', which uses Microsoft Research's Simple Encryption Arithmetic Library (SEAL), to enable learning models to easily operate on encrypted data [13].

D. Sharing Sensitive Information

It is a very common occurrence nowadays. You search 'product X' on your browser, and suddenly you see advertisements for 'product X' and other related products on Google or Social Media. Google and social media sites like Facebook and Instagram have become somewhat of an example for those who say that our online privacy is being continually eroded. These sites store personal information, location history, likes, dislikes and keep track of what their users look at while logged onto the platform, whether this be on the platform itself or even on other browser sessions, to serve up personalised advertisements.

Picture a scenario: Person X has been to Cafe Y three times in the past month, an advertising services know this from their location history. Person X is nearby Café Y and the advertising service knows this by their live location. The person is sent a coupon for Café Y by the agency.

This development may seem like a win-win for some; Café Y can target someone who is more likely to buy their product, and Person X receives advertisements for things that they like and saves money. The problem to some is the invasion of privacy that these services entail. FHE would allow this contextual information, i.e. Person X's location, to be encrypted by them using their own key, before being sent to the advertisement service. The service could then use this to decide what ads to serve. If we take the earlier example, this will allow Person X and Café Y to both get their win without Person X giving up personal information about themselves in a readable form that could be used for malicious purposes. A person could even upload sensitive financial material such as an income statement to calculate their taxes and still have peace of mind that their information is essentially invisible to the other party [5].

VII. Coordinated Efforts Towards Adoption

The amazing advances possible by FHE would not count for much if there was not a willingness from the major players in different industries to adopt this new technology. In 2017 Microsoft, IBM and Duality Technologies co-founded the 'HomomorphicEncryption.org' group [14]. This group describes itself as "an Open Industry/Government/Academia Consortium to Advance Secure Computation" and maintains a community standard for homomorphic encryption. The group advocates for RLWE FHE due to the confirmed the hardness of the RLWE problem from what it calls a "a long line of peer-reviewed research" and its ability to ensure post-quantum security. This group organises workshops, publishes white papers, and hopes to achieve a standard that is easy for non-experts in the field to use. The website contains a list of different FHE techniques implemented in various programming languages such as the previously mentioned HELib from IBM and Microsoft's SEAL library. An impressive list of big industry, government and academia players have participated in their recent standardisation workshops such as Google, SAP, NIST, the ITU and over 20 academic institutions from across the world. Before organisations are willing to make large investments betting on new technologies such as FHE schemes, standardisation is always an important step forward and so these recent movements are an encouraging sign for the future of FHE.

VIII. Conclusion

Homomorphic Encryption has the potential to solve many of the privacy issues which are emerging from the growing use of cloud computing and big data. It also has the potential to secure our information from the seemingly inevitable paradigm shift that will be brought about by quantum computing. First dreamed up in 1978, but only made a reality in 2009, homomorphic encryption has become a hot-topic in the world of cryptography. After Gentry, Sahai and Water revealed their GSW approach to FHE in 2013, Gentry estimated that efficient FHE could become a reality within the next decade [9]. Whether this estimation holds to be true remains to be seen as research and investment into developing practical and scalable FHE is ongoing. In May last year a team made use of Trusted Encryption Environments (TEEs) to move the bootstrapping process to a SGX enclave which resulted in a four orders of magnitude speed up to the process [3]. There is also clear interest from major tech companies, government agencies and academic institutions to begin formally standardising FHE techniques. Many of these groups are conducting ongoing research into the field, while others have released packages to put the technology into the hands of people. The quantum-secure nature of current FHE is also of massive importance. The field of cryptography can be slow to accept new techniques so perhaps implementing homomorphic encryption techniques could serve as a way to simultaneously future-proof against post-quantum cryptography given that the hardness of current FHE schemes stands up to this new adversary. Little over a decade has passed since Gentry's seminal paper of FHE was published. While there is still some ways to go towards making FHE fully scalable and comparably efficient to other cryptographic techniques; software and hardware based improvements, coupled with a strong interest from important players makes homomorphic encryption a strong candidate to secure the present and future of our information online.

References

[1] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of Secure Computation, Academia Press*, pp. 169–179, 1978.

[2] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford, CA, USA, 2009.

[3] R. Frederick, "Core concept: Homomorphic encryption," *Proceedings of the National Academy of Sciences*, vol. 112, no. 28, pp. 8515–8516, 2015. [Online]. Available: https://www.pnas.org/content/112/28/8515

[4] B. Vankudoth and D. Vasumathi, "Query based computations on encrypted data through homomorphic encryption in cloud computing security," 03 2016, pp. 3820–3825.

[5] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?" in *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop*, ser. CCSW '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 113–124. [Online]. Available: https://doi.org/10.1145/2046660.2046682

[6] D. J. Wu, "Fully homomorphic encryption: Cryptography's holy grail," *XRDS*, vol. 21, no. 3, p. 24–29, Mar. 2015. [Online]. Available: https://doi.org/10.1145/2730906

[7] J. Desjardins, "How much data is generated each day?" [Online]. Available: https://www.weforum.org/agenda/2019/04/how-much-data-is-generated-each-day-cf4bddf29f/

[8] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ser. ITCS '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 309–325. [Online]. Available: https://doi.org/10.1145/2090236.2090262

[9] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Advances in Cryptology – CRYPTO 2013*, R. Canetti and J. A. Garay, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 75–92.

[10] P. Chaudhary, R. Gupta, A. Singh, and P. Majumder, "Analysis and comparison of various fully homomorphic encryption techniques," *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, pp. 58–62, 2019.

[11] C. S. Division, I. T. Laboratory, N. I. of Standards, Technology, and D. of Commerce, "Post-quantum cryptography." [Online]. Available: https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

[12] A. El-Yahyaoui and M. D. E.-C. El Kettani, "Fully homomorphic encryption: Searching over encrypted cloud data," in *Proceedings of the 2nd International Conference on Big Data, Cloud and Applications*, ser. BDCA'17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3090354.3090364

[13] "Microsoft seal: Fast and easy-to-use homomorphic encryption library." [Online]. Available: https://www.microsoft.com/en-us/research/project/microsoft-seal/

[14] "Homomorphic encryption standardisation." [Online]. Available: https://homomorphicencryption.org/introduction/