2019 International Conference on Computing, Power and Communication Technologies (GUCON)
Galgotias University, Greater Noida, UP, India. Sep 27-28, 2019

Analysis and Comparison of Various Fully Homomorphic Encryption Techniques

Pratibha Chaudhary
Amity University
Uttar Pradesh
pratibha.chaudhary4991@gmail.com

Ritu Gupta
Amity University
Uttar Pradesh
ritu4006@gmail.com

Abhilasha Singh
Amity University
Uttar Pradesh
abhilashasingh28@gmail.com

Pramathesh Majumder
Amity University
Uttar Pradesh
pramathesh@outlook.com

Abstract— **Calculations can be carried out on encrypted form of data- is the essence of homomorphic encryption. Homomorphic encryption has resolved the security issues for storing data on the third-party systems (e.g. cloud or untrusted computer, service providers etc.). Most significant category of homomorphic encryption is fully homomorphic encryption. It permits unbounded number of operations on the encrypted form of data and output by system is within the ciphertext space. This paper provides the essentials of Homomorphic encryption and its various classifications i.e. partially homomorphic encryption, somewhat homomorphic encryption and fully homomorphic encryption. Mainly emphasizes on full homomorphic encryption and an investigation of different full homomorphic encryption schemes that uses the hardness of Ideal-lattice, integers, learning with error, elliptic curve cryptography based.**

Keywords— homomorphic encryption (HE), partial homomorphic encryption (PHE), somewhat homomorphic encryption (SWHE), full homomorphic encryption (FHE), third-party, elliptic curve cryptography (ECC).

I. Introduction

In the world of IT, the data is generally stored and processed on third-party (cloud or untrusted computer, service providers etc.). Due to this, two major concerns arise. First, the legacy encryption schemes don't provide the security of data at third- party (i.e. stores unencrypted data). Third-party may misuse the data. Second, to perform any kind of operations on encrypted data first it is decrypted. These concerns may lead to the discovery of homomorphic encryption that permits performing calculations on data in encrypted form which is stored on third-party without decrypting it. The idea of performing computation over encrypted data called privacy homomorphism [1] came in 1978. In the same year 1978, RSA was also introduced by them. RSA [1] is asymmetric encryption scheme i.e. have open key (i.e. public) for encrypting and confidential key (i.e. secret) for decrypting known only to authorized user. RSA allows only multiplication operations on encrypted data. The homomorphic encryption scheme that allows both add and multiply unbounded operations on encrypted data called fully homomorphic encryption scheme. Fully homomorphic encryption scheme (FHE) was an open problem. Many researchers proposed their schemes that were El-Gamal scheme [2] but works for multiplication operations, Paillier scheme [3] but works for addition operations etc.

First fully homomorphic encryption scheme from somewhat homomorphic encryption scheme was given by Gentry [4], a noteworthy leap forward. This was based on the hardness ideal lattices [5].

Homomorphic encryption has numerous useful applications. For instance, Paillier employed in online voting system and threshold scheme [6], RSA is used to secure internet, Banking and credit card transaction [6], El-Gamal is used in hybrid systems, also used in secure multi-party computation [6] and so on.

II. Literature Survey

In this paper, we will investigate the previous researches done in the field of cryptography related to homomorphic encryption. Study and analyze the various homomorphic encryption schemes. Provide the insight of the best category of homomorphic encryption called fully homomorphic encryption and its various categories. Homomorphic encryption concept is taken from abstract algebra. Considering plaintext $y1, y2 \dots yn$ and Encryption scheme E if it follows:

A. *Additive property:*

$$(y1 + y2 + y3 \dots\dots + yn) = (y1) + E(y2) + E(y3) \dots + E(yn) \qquad (1)$$

B. *Multiplicative property:*

$$(y1 * y2 * y3 \dots * yn) = (y1) * E(y2) * E(y3) \dots E(yn) \qquad (2)$$

Then it is homomorphic.
For creating any kind of homomorphic encryption scheme, we use combination of addition and multiplication Boolean circuits (AND and XOR gate). Consider homomorphic encryption scheme for Asymmetric encryption is divided into four algorithms as follows:

$$H = (generating\ key, encrypt, evaluate, decrypt) \quad (3)$$

The running time of all these algorithms depends on security parameter λ.
a). Creating key Algorithm:
It takes security parameter λ as input i.e. number of bits in keys and output secret key s and public key p.
b). Encrypt Algorithm:
It takes plaintext $y1, y2 \dots yn$ as input. It divides plaintext into bits (0,1) $b1, b2 \dots\dots bi$ and output ciphertext $c1, c2, \dots$

ci.

$$c = Encrypt\,(p, bi) \quad (4)$$

c). Evaluate Algorithm:

It takes p and function fun as input and output new ciphertext $c1, c2 \dots cj$.

$$c = Evaluate(p, fun, ci) \quad (5)$$

d). Decrypt Algorithm:

Takes ciphertext $c1, c2 \dots cj$ and s as input and output the plaintext $y1, y2 \dots yn$.

$$yn = Decryp(s, cj) \quad (6)$$

The homomorphic encryption is categorized into three class as shown in Fig. 1 that are:

1. *Partially homomorphic encryption (PHE):*

It allows either addition or multiplication operations i.e. only one kind of any number of operations. For example – Unpadded RSA, El-Gamal etc.

2. *Somewhat homomorphic encryption (SWHE):*

It allows add and multiply operations, but number of operations are limited. For example – BGN [7] etc.

3. *Fully homomorphic encryption (FHE):*

It allows arbitrary number of add and multiply operations. For example – lattice-based [5], integer based [10], learning with error based [14] etc.

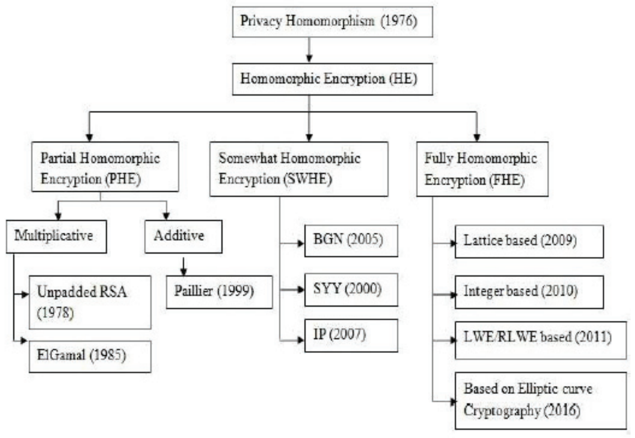
Privacy Homomorphism (1976)
Homomorphic Encryption (HE)
Partial Homomorphic Encryption (PHE)
Somewhat Homomorphic Encryption (SWHE)
Fully Homomorphic Encryption (FHE)
Multiplicative
Additive
Unpadded RSA (1978)
ElGamal (1985)
Paillier (1999)
BGN (2005)
SYY (2000)
IP (2007)
Lattice based (2009)
Integer based (2010)
LWE/RLWE based (2011)
Based on Elliptic curve Cryptography (2016)

Fig. 1. Classification of Homomorphic encryption schemes

III. Fully Homomorphic Encryption Schemes

Fully Homomorphic Encryption is a type of cryptosystem that encrypts the ciphertext arbitrarily. It allows the programs to run on any type of functionality that supports the encryption system.

A. *FHE scheme formed using Ideal Lattice (2009)*

Bootstrappable encryption scheme is created using ideal lattices [5] because:

- Decryption circuit complexity of algorithms is very low in typical lattice-based than the schemes like unpadded RSA or ElGamal.
- Addition and multiplication property of homomorphism is provided by ideal lattices.

Gentry's construction involves three steps that are:

- Construction of somewhat homomorphic encryption: Constructing to evaluate low-degree polynomials for a limited number of ADD and MULT operations. A ciphertext is of form

$$c = u + n \quad (7)$$

Where u is vector in the ideal lattice [5] and n is an "error" or "offset" or "noise" vector attached with plaintext m. The error component is increasing by applying each homomorphic operation it reaches a value that is inevitable which at a certain point does not allow ciphertext decryption correctly anymore.

- Squashing: Squashing is used to minimize the depth of decryption circuit as it increases to higher value. SWHE scheme Decryption circuit works well with small depth.
- Bootstrapping: Bootstrapping is a process of refreshing the ciphertext to reduce noise. If a somewhat homomorphic encryption scheme can assess itself (augmented) decrypting circuit, then it is said to be bootstrappable.

B. *Fully Homomorphic Encryption without using Bootstrapping:*

In 2010, Gentry et. al. proposed the idea of fully homomorphic encryption schemes using bootstrapping mechanisms. This allowed to use any number of additions to the original data but with only single multiplication.

In 2013, Wei et. al. presented the idea that allowed any number of additions and multiple number of multiplications on the original data.

Symmetric Learning based Fully Homomorphic Encryption scheme:

Later, Symmetric learning mechanism based fully homomorphic encryption scheme was proposed that was mainly based on machine learning techniques to learn about the encryption pattern. This method excluded the use of bootstrapping and used machine learning approach instead. This technique used dimension-reduction mechanisms which allowed shortening the original text and reduced the complexity of decryption mechanisms which doesn't affect the additional assumptions.

Ring Fully Homomorphic Encryption schemes:

Following SFHE scheme, a new FHE scheme was developed – RFHE (Ring Fully Homomorphic Encryption). This scheme didn't used noise reduction mechanisms and concerned a non-octonion ring over the noise reduction.

Later, Liu proposed an idea of symmetric fully homomorphic encryption scheme extending the RFHE scheme, which was depended on non-commutative ring fields. This idea also excluded the possibility of bootstrapping and allowed a greater number of additions and multiplications on the original data. It didn't used noise reduction and was dependent on approximate-GCD mechanisms.

C. Implementation of Fully homomorphic encryption scheme (2011)

For implementing fully homomorphic encryption, Gentry and Halevi [8] used variant similar to Smart and Vercauteren [9]. Smart and Vercauteren were unable to implement bootstrapping functionality when converting somewhat homomorphic encryption to fully homomorphic encryption. The optimization includes bootstrapping functionality; a batching technique for encryption etc is done in all aspects to implement fully homomorphic encryption. In Somewhat homomorphic encryption, key generation does not require full polynomial inversion [8].

TABLE I. SHOWS LATTICE DIMENSIONS AND SETTING IN DIMENSIONS

Lattice dimensions	**Setting in dimension (Size)**
Toy	$512 = 2^9$
Small	$2048 = 2^{11}$
Medium	$8192 = 2^{13}$
Large	$32768 = 2^{15}$

Table 1 above shows the lattice with several dimensions taken into consideration for implementing FHE.

TABLE II. PUBLIC KEY SIZE RANGES

Lattice dimensions	**Public key size**	**One Bootstrapping operation runtime**
Small	70 Megabytes	30 seconds
Large	2.3 Gigabytes	30 minutes

Table 2 above shows the public key size ranges and one bootstrapping operation runtime used for small and large lattice dimensions. The One Bootstrapping operation runtime of 1-CPU 64-bit machine with large memory [8] isconsidered.

D. FHE scheme formed using integers (2010)

Van Dijk et al [10] came up with fully homomorphic encryption scheme using Gentry's construction [5] but defined over integer instead of ideal lattices because they are conceptually simpler to show that thecomplicated fully homomorphic encryption scheme can be accomplished using "basic" techniques. The scheme used hardness of the approximate integer greatest common divisors (approximate GCD) problem [11] was analyzed by Howgrave-Graham.

General construction:

- Creating key Algorithm: An odd integer key , $k \in [2n-1, 2n)$.
- Encrypting Algorithm$(k,\ x\ )$: For encrypting a bit $x \in \{0,\ 1\}$, set ciphertext an integer value whose residue mod p has the similar parity as the plaintext. Like set $c = k * q + 2r + x$, where the integers q, r are chosen at random such that $2r$ is less than $k/2$ in absolute value.
- Decrypting Algorithm $(k,\ c)$: $x = Output\ (c\ mod\ k)\ mod\ 2$.
- Constraint: $x + 2r < k/2$ and choose r: $r \approx 2\sqrt{n}\ and\ q \approx 2n\text{^}3$

Homomorphism over addition:

$$(x1) + E(x2) = x1 + 2r1 + kq1 + x2 + 2r2 + kq2 = (x1 + x2) + 2(r1 + r2) + (q1 + q2)$$

$Nois = (x1 + x2) + 2(r1 + r2)$, it grows linearly as shown in Fig. 2(a).

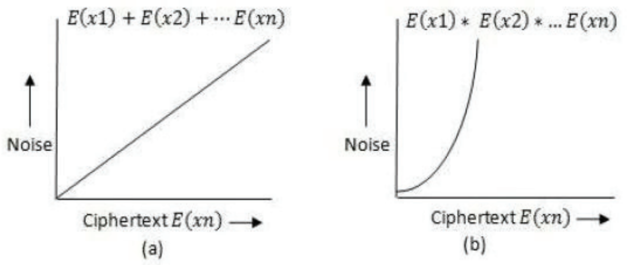

Fig. 2. Noise expansion with encryption

Homomorphism over Multiplication:

$$(x1) * E(x2) = (x1 + 2\,r1 + kq1)(\,x2 + 2\,r2 + kq2) = x1\,x2 + 2(x1r2 + x2r1 + 2\,r1\,r2) + (kq1\,q2 + 3q1r2 + x1q2 + x2q1)k \quad (9)$$

$Nois = x1\ x2 + 2(x1r2 + x2r1 + 2\ r1\ r2)$, it grows exponentially as shown in Fig.2 (b).

This is a symmetric scheme, but easily converted into asymmetric scheme. The public key is formed using "encryptions of zero", particularly integers

$$p = qi * k + 2ri \quad (10)$$

Where qi, ri are described above, k is private key. Same decryption is used for decrypting the ciphertext. As no efficient algorithm here exists, for recover k from the given pis in polynomial time, therefore the scheme is considered as secure.

E. FHE scheme formed using learning with error (2011)

Learning with Error (LWE) is one of the hardest problems to solve in practical time for even post-quantum algorithms as mentioned [12]. Regev had first introduced "learning from parity with error" problem [13] that reduced worst case hardness of lattice problems like shortest vector problem (SVP) [14] to learning with error (LWE) problem. Brakerski and Vaikuntanathan came up with the scheme which uses ring learning with error (RLWE) [14]. RLWE used polynomial- LWE (PLWE). For understanding hardness assumption of RLWE, for instance, take a polynomial samples over a ring having form $(ri, rik + yi)$, where k is a random "secret ring element", $ri's$ are uniformly random in the ring, and yi are "small" elements of ring, an eavesdropper cannot distinguish random pairs of ring elements in this sequence of samples [14]. This is further decreased to worst case problem of SVP. This FHE scheme was also constructed from SWHE using Gentry's bootstrapping and squashing techniques [5]. To avoid squashing in this scheme, decrease to worst case hard problems at the cost of depending on a "sparse" version of LWE [14]. The alternate of squashing is re-linearization. Re- linearization reduced the ciphertext size. No generation of lattice basis required for generating a key. The RLWE scheme is circular secure that means it permits to safely encrypt its self-secret key i.e. ring element. RLWE used for practical applications more than LWE as it is more efficient. RLWE is circular secure (key-dependent security) with respect to linear functions of secret key.

F. FHE formed using elliptic curve cryptography (2016)

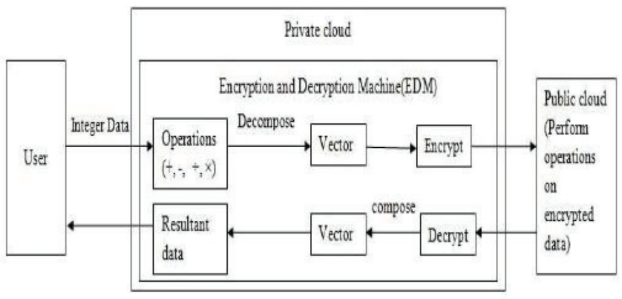

Fig. 3. Homomorphic encryption using elliptic curve cryptography

Figure 3 above shows the step involved in homomorphic encryption using elliptic curve cryptography. User wants to process data, first it goes to private cloud. Private cloud has the Encryption and Decryption Machine (EDM) that is responsible for encrypting and decrypting the data. Public cloud performs the computational operations on the data in encrypted form and returns the obtained result backward to the EDM. EDM will provide the required resultant data back to user.

IV. RESULTS

Our goal is to achieve a practically feasible FHE scheme that is unrestricted. One example of practical implementation for FHE schemes is provided as follows:

1. Lattice- based scheme is implemented on IBM system ×3500 server [8] and 6 Intel xeon 2.4 GHz processor [16] using language C++ and NTL library.
2. Integer-based scheme is implemented on SAGE software on Intel core i5, 3.30 GHz x4, 8 Gb RAM [17].
3. RLWE is implemented in HELib using C++ language and a mathematical library NTL [18].
4. Elliptic curve cryptography based HE is empirically evaluated and test on GPS data [15].

TABLE III: COMPARISON OF FULLY HOMOMORPHIC ENCRYPTION SCHEME

Property	**Ideal lattice based**	**Integer based**	**Ring learning with error**	**Elliptic curve cryptography based**
Introduced in year	2009	2010	2011	2016
Hardness	Closest vector problem, sparse subset sum problem	Approximate GCD problem	Rings learning with errors	Discrete logarithm of an arbitrary elliptic curve component concerning to known base point is impractical
Security	More	Less	More than Ideal based	Most
Simplicity	Less	Most	Simpler than Ideal based	More
Limitations	Public key size is too large	Reduced security	NA	Does not support floating point calculation
Efficiency	Efficient	Least efficient	More efficient than ideal lattice based	Most efficient

After critically studying and analyzing the above mentioned fully homomorphic encryption schemes, the comparison is done.

Some properties like hardness, introduced year, security of scheme, efficiency etc. are identified while comparing and the result of comparison are listed in Table 1.

V. CONCLUSION AND FUTURE WORK

Homomorphic encryption existed from 30 years still is open problem for research. Various Homomorphic encryption schemes came have a scope of improvement.

The improvement can be done in any aspect like improving generation of key or encryption, decryption etc. The implementation cost is very high, and complexity of homomorphic encryption scheme is also high. It can be reduced in future. In this paper, we have classified Homomorphic encryption scheme and mainly focused on various types of FHE. FHE schemes formed using lattice, integer, ring learning with error and elliptic curve cryptography based is implemented in real-life applications. The example of each is provided above. FHE is not practically implemented on every platform. The mentioned FHE schemes supports only single user setting notwithstanding with applications that take multiple data from different users [19].

References

[1] Rivest, R. L., Adleman, L., & Dertouzos, M. L. (1978). On data banks and privacy homomorphisms. Foundations of secure computation, 4(11), 169-180.

[2] ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory, 31(4), 469-472.

[3] Paillier, P. (1999, May). Public-key cryptosystems based on composite degree residuosity classes. In Eurocrypt (Vol. 99, pp. 223-238).

[4] Gentry, C. (2009). A fully homomorphic encryption scheme. Stanford University.

[5] Gentry, C. (2009, May). Fully homomorphic encryption using ideal lattices. In STOC (Vol. 9, No. 2009, pp. 169-178).

[6] Parmar, P. V., Padhar, S. B., Patel, S. N., Bhatt, N. I., & Jhaveri, R. H. (2014). Survey of various homomorphic encryption algorithms and schemes. International Journal of Computer Applications, 91(8).

[7] Boneh, D., Goh, E. J., & Nissim, K. (2005, February). Evaluating 2-DNF Formulas on Ciphertexts. In TCC (Vol. 3378, pp. 325-341).

[8] Gentry, C., & Halevi, S. (2011, May). Implementing Gentry's Fully-Homomorphic Encryption Scheme. In EUROCRYPT (Vol. 6632, pp. 129-148).

[9] Smart, N. P., & Vercauteren, F. (2010, May). Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. In Public Key Cryptography (Vol. 6056, pp. 420-443).

[10] Van Dijk, M., Gentry, C., Halevi, S., & Vaikuntanathan, V. (2010, May). Fully homomorphic encryption over the integers. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 24-43). Springer Berlin Heidelberg.

[11] Howgrave-Graham, N. (2001, March). Approximate integer common divisors. In CaLC (Vol. 1, pp. 51-66).

[12] Acar, A., Aksu, H., Uluagac, A. S., & Conti, M. (2017). A Survey on Homomorphic Encryption Schemes: Theory and Implementation. arXiv preprint arXiv:1704.03578.

[13] Regev, O. (2009). On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM), 56(6), 34.

[14] Brakerski, Z., & Vaikuntanathan, V. (2011, August). Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Annual cryptology conference (pp. 505-524). Springer, Berlin, Heidelberg.

[15] Hong, M. Q., Wang, P. Y., & Zhao, W. B. (2016, April). Homomorphic Encryption Scheme Based on Elliptic Curve Cryptography for Privacy Protection of Cloud Computing. In Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on (pp. 152-157). IEEE.

[8] Smart, N. P., & Vercauteren, F. (2014). Fully homomorphic SIMD operations. Designs, codes and cryptography, 1-25.

[9] Gerasimov, A. N., Epishkina, A. V., & Kogos, K. G. (2017, February). Research of homomorphic encryption algorithms over integers. In Young Researchers in Electrical and Electronic Engineering (EIConRus), 2017 IEEE Conference of Russian (pp. 398-403). IEEE.

[10] Halevi, S., & Shoup, V. (2013). Design and implementation of a homomorphic-encryption library. IBM Research (Manuscript), 6, 12-15.

[11] Tourky, D., ElKawkagy, M., & Keshk, A. (2016, October). Homomorphic encryption the “Holy Grail” of cryptography. In Computer and Communications (ICCC), 2016 2nd IEEE International Conference on (pp. 196-201). IEEE.