World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:9, No:3, 2015

Big Data Analytics and Data Security in the Cloud via Fully Homomorphic Encryption

Victor Onomza Waziri, John K. Alhassan, Idris Ismaila, Moses Noel Dogonyaro

***Abstract*—**This paper describes the problem of building secure computational services for encrypted information in the Cloud Computing without decrypting the encrypted data; therefore, it meets the yearning of computational encryption algorithmic aspiration model that could enhance the security of big data for privacy, confidentiality, availability of the users. The cryptographic model applied for the computational process of the encrypted data is the Fully Homomorphic Encryption Scheme. We contribute a theoretical presentations in a high-level computational processes that are based on number theory and algebra that can easily be integrated and leveraged in the Cloud computing with detail theoretic mathematical concepts to the fully homomorphic encryption models. This contribution enhances the full implementation of big data analytics based cryptographic security algorithm.

***Keywords*—**Data Analytics, Security, Privacy, Bootstrapping, and Fully Homomorphic Encryption Scheme.

I. Introduction

BIG Data Analytics as defined in [21], is now the pivotal for big data processing and Analytics for critical enterprises and governmental institutions application. Hence there is an accelerating need for the development of Big Data infrastructure that will support storage and processing of big data in cloud computing environment.

Cloud Computing is now the tool of choice for big data processing and analytics due to its reduction in cost, broad network access, elasticity, resource pooling, and measured service [13]. The main feature of Cloud computing to consumers is that it allows the storing and analyzing the sharing of computing resources while it handles the fluctuations in the volume and velocity of the data. Despite the soundtracks of Cloud Computing, it also comes with risks.

A. Cloud Security

Cloud Computing is an Information Technology (IT) model for computing. The IT components consist of the critical infrastructure: hardware, software, networking, and Services. These are necessary for the development and delivery of cloud services via the Internet or private network [10].

As we look critically with in-depth criticism on the above definition, no idealization of security is mentioned, this translates to mean that cloud competing is lacking in security structure, confidentiality, Integrity and availability. However, to provision Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) is not adequate enough if the Cloud Providers do not give maximum assurance of good enough security for privacy and confidentiality of both customer's data and data being stored or computed.

The world is full of security challenges to which can be either human security or the data stored in the Internet. Hence, cloud computing critical infrastructures need the protection of the cryptographic abstracted TRIAD-confidentiality, integrity and availability; which are the basis of information security. There is the need to secure the cloud for securing the Cloud means to have the treatment (computational processes) and storage (databases hosted by the cloud providers) free from attacks by an adversary [20].

Both models of cloud computing architectures like the public clouds, private clouds and hybrid Cloud need real-time security for the data stored in them. Besides, data on transmission over the public Internet need security if the clients must meet and need services from the cloud providers. It therefore presages that nearly all computing accessory of data in transmission demand encryption based on worst-case hardness or asymptotically secure algorithms. However, performing computations on the encrypted stored data in the cloud pose some complex challenges as the private key will have to be provided to the cloud providers to decrypt the encrypted data before meaningful computational analysis could be carried out on the data. This would make critical information available to the providers; who may turn out to be an adversary. Hence, there are numerous challenges to the data confidentiality and the user's privacy. Furthermore, the application of Data mining and other Data analytics algorithms need to be efficiently encrypted. For with no clear useful algorithm to support encrypted data computation in the cloud, will prove an inflexible far distant paradigm to achieve. The only ingenious proposal here is to have the database encrypted before sending it to the cloud providers and for the cloud computing functions or vendors to compute the data for the client as may be requested; and without exposing the original data. The computational process is achievable by applying homomorphic encryption scheme either as fully homomorphic encryption (FHE) or Somewhat Homomorphic

V. O. Waziri is an Associate Professor of Cyber Security Science with the Department of Cyber Security; he is the pioneer Head of Department, School of ICT, Federal University of Technology, Minna-Nigeria (Corresponding Author; mobile: +234806351893; e-mail: victor.waziri@futminna.edu.ng b or onomzavictor@gmail.com).

J. K. Alhassan is the Current Head of Cyber of Cyber Security Science Department of the Federal University of Technology,

Ismaila Idris is PhD graduate in Computer Science.

M.N. Dogonyaro is a Master Student in Cyber Security Science in the Federal University of Technology, Miina. He is also the System Analyst in the Department of Cyber Security Science of the Federal University of Technology, Miina

Encryption (SHE); as we discuss in the next subsequent sections of this paper.

B. Big Data Analytics Security

References [16] and [22] view the term big data as information technology that defines a collection of data sets so large and complex that it becomes very difficult to processing using on-hand database management tools or traditional data processing applications. Big data analytics has numerous challenges that include capture, processing, storage, search, sharing, analysis, and visualization.

Big data is often characterized with Vs dimensions. These include Volume, Velocity, Variety, and Veracity. Reference [23] gives details of each dimension as follow:

1. **Volume:** The term Volume explains that data Volume typically ranges from multiple terabytes to zetabytes monthly. This certainly fits the enterprise security model as it is not uncommon for large organizations to collect tens of terabytes of security data on monthly basis
2. **Velocity:** This term is defined with respect to real-time data analyst requirements. In cybersecurity, velocity can refer to the need for immediate anomaly, or rapid incident response in real-time. Real-time data analysis is critical here to minimize damages associated with cybersecurity attack.
3. **Variety:** This term explains big data to be made up of multiple data type requirements and feeds that include structured and unstructured data. From a security consideration, data variety could include big files, network flows, IP packets capture external threat/internal threat/vulnerability, intelligence; click streams, network/physical access, and social networking activity, etc. It is therefore not unusual for enterprises to collect hundreds of different types of data for security analysis using various algorithmic models.
4. **Veracity:** The term simply means that big data must be trustworthy and accurate. From the security insight or perspective, this means that trusting the confidentiality and availability of data sources like log files and external feeds.

These outstanding definitions of the big data Vs dimensions clearly give perceptive that big data should be made secure through diversified tools of Cryptography. The Vs further signify that encrypted data should be analyzed wherever the client stores them without decryption so as to meet the security and privacy of the client. Such computing models should be well structured in development based on worst-cases hardness or should be asymptotically secure. Computational Security algorithms abound in the cryptographic field; but the most significant of them for big data implementation is the Fully Homomorphic Encryption schemes that took insightful discovery [4] PhD thesis; also in [19]. This algorithm, though complex in current implementation, has opened the gateway for implementable algorithm that are already in the offing. However, more efficient algorithms developed to enhance [4] such as in [5], [6] and [18]. Before this time [4], [14] had conceived the idea of homomorphic encryption which many partial homomorphism that existed before [4] model was developed. More details on these homomorphic paradigms in some sections below.

C. Our Proposal

Our works is not developing a new model but a cursory survey that can blend the Big Data Analytics, Cloud Computing and the Fully Homomrphic Encryption as defined [11] and [16] schemes to enhance security. Thus, our contribution is to unify these already existing research entities as studied separately into some meaningful research exposition.

D. Organization

The rest of the paper is structured as follows: In Section II, we overview the development of classical public Key Encryption scheme and Probabilistic encryptions. Also useful definitions of security such as CPA, CCA 1 and CCA2 are provided. In Section III, we provide Partially Homomorphic Public Key Encryption scheme with adequate examples of Homomorphic encryption Schemes, Section IV reviews the Fully Homomorphic Encryption Scheme with emphasis on [4]. Section V is based on the Application of Fully Homomorphic Scheme in the Cloud Computing platform. The last section, Section VI, gives an explicit conclusion with future works recommendation. References form the last non-itemized section.

II. Public Key Encryption in a Nutshell

The modern advancements in the utilization of different communications systems for computing environments are changing rapidly by the hours. Today, almost all individuals have portable computing devices (this could be in form of phones or any other computing devices with the Internet connection) and these can access servers in the cloud environments. In recent computing scenarios, clients are trusted (but weak) [18], while computationally strong servers are not trusted due to the clients' lack of control over their operational functionalities. The standard improvement to this problem is to encrypt the pieces of information (data); this perfectly solves any privacy issues [18]. Data could be encrypted using symmetric encryption scheme or asymmetric encryption scheme or called public key encryption, we outline the later here [9] outlines the public encryption scheme that permits the sharing of asymmetric keys for encryption and transmit the encrypted data over the public unsecure communication channel. Originally, Public key was invested by [3], called public key or asymmetric key. The scheme was called Diffie-Hellman Encryption scheme. The security on this type of scheme is based on hard problems in mathematical constructs, which is assumed difficult to solve in polynomial time. Developed [24] the public key algorithm called the RSA. In [14] proposed the additive and multiplicative privacy homomorphism, respectively. These computational algorithmic functions, however, did not by themselves; provide the chosen plaintext attack (CPA) security.

Nonetheless, [25] model that was derived based on exponentiation function is homomorphic and CPA-secure. In a characteristic and succinct formulation; and as adduced by [9], let us shortly look into the theoretical construction of the public key encryption scheme before further discourse on partial homomorphic encryption model and fully homomorphic encryption scheme as developed by [4] PhD Thesis. This generic syntax of public-key encryption scheme runs as follow:

A public-key encryption scheme is a 3-tuple of probabilistic, polynomial-time algorithms (Gen, Enc, Dec) that satisfies the following:

1. Algorithm Gen takes as input a security parameter "1^{λ}"in unary form and outputs the plaintext space M, the ciphertext space C, a pair of keys (pk, sk) which are the public and secret keys respectively. That is $(PK, SK) \leftarrow G(1^{\lambda})$. It is assumed that both pk and sk have length of at least λ, and that n can be determined from pk, sk.
2. The algorithm Enc takes as input a public key pk and a message M from underlying plaintext space (that may depend on pk). It outputs a ciphertext, and this is written as $C \leftarrow Enc_{pk}(m)$. In clearer exposure, it's $C \leftarrow Enc(pk, M)$.
3. The Decryption algorithm M takes as input a private key sk and a ciphertext c, and outputs a message M or a special symbol $\perp$ denoting failure. It is assumed without loss of context that Dec is deterministic, and we write this as $m := Dec_{sk}(C)$ or $M \leftarrow Dec(sk, C)$

In furthering with our building the encryption schemes, we need to consider the Probabilistic Public-Key Bit-Encryption scheme). The first public-key cryptosystems (such as the RSA invented by [23] were considered deterministic algorithms that were based on trapdoor functions. Deterministic functions are functions that are easy to compute one-way but are difficult or hard to invest-unless some trapdoor information is known [27]. As a trapdoor function, the encrypted messages can only be decrypted by the legal receiver who has the trapdoor details known as the private key. The trapdoor has its weaknesses according to [7], the two main draw backs of the encryption schemes based on trapdoor functions are:

1. Inverting may be easy for plaintexts of some special form; and
2. It could be easy to compute at least partial information of the plaintexts

Besides, using the deterministic scheme, it is easy to detect if the message is sent twice. These offered points, inspired the development of probabilistic public-key encryption schemes [8].

Definition 1 (Probabilistic Public-Key Bit-Encryption Scheme): The Probabilistic Public-Key Bit-Encryption Scheme is triple-tuple $(\mathrm{K}, \mathrm{E}, D)$ with security parameter n that consists of:

a. *K, the key generator: A probabilistic algorithm that on input n, outputs a pair (e, d), where e is the public key and d is the private key.*
b. *E, the encryption function, three inputs: the public key e, the plaintext bit $b \in \{0, 1\}$, and a random string r of length $p(n)$ for some polynomial $p(.)$. We write the ciphertext encryption algorithm as $Een_e(b, r)$.*
c. *D, the decryption function, with two inputs: the private key d and the ciphertext c. The decryption algorithm* is *written as $D_d(c)$.*

Decryption of any encryption of a bit yields the encrypted bit, that is

$$\forall n \in N \forall Ie, d) \in K(1^n) \forall b \in \{0, 1\}$$
$$for\ all\ r \in \{0, 1\}^{p(n)} : D_d((Enc_{\in}(b, r))) = b$$

A. The Quadratic Residuosity Problem

In this subsection, we account for the number-theoretical results that underlies the [8] encryption scheme. This gives credence to this relational expression $Z_n^n := \{a \in N \mid 1 \le a \le n-1 \Lambda\ \gcd(a, n) - 1\}$, which is the multiplicative group of Z_n. That is to say that all numbers less than n that have multiplicative group inverses modulo n.

Definition 2 (Quadratic Residues): An element $a \in Z_n^*$ is said to be quadratic residue (square) modulo n if there exists an integer $x \in Z_n^*$, such that $x^2 \equiv a(\mathrm{mod} n)$. Every such x is called a square root of 'a' modulo n. If no such x exists, then 'a' is called a quadratic non-residue modulo n.

We denote the set of all quadratic residues modulo n by Q_n, and the set of all non-quadratic non-residue by $\overline{Q}_n$.

Lemma 1: Let p be a prime. Then $|Q_p| = |\overline{Q}_p| = \frac{1}{2}|Z_p^*| = \frac{p-1}{2}$; where p and q are two odd primes, such that $n := pq$, $a \in Z_n^*$ Then, $a \in Q_n$ if and only if $a \in Q_p$ *and* $a \in Q_q$. Thus for $n = pq$, we have $|Q_n| = \frac{1}{4}(p-1)(q-1)$.

Definition 3 (Legendre Symbol): Let p be an odd prime, 'a' an integer, such that $\gcd(a, p) = 1$. The Legendre Symbol is defined as:

$$\left(\frac{a}{p}\right) = \begin{cases} 1, & if\ a \in Q_p \\ -1 & if\ a \in \overline{Q}_p \end{cases}$$

Lemma 2: Let p represents an odd prime, $a, b \in Z_p^*$. Then, $\left(\frac{a}{p}\right)\left(\frac{b}{p}\right) = \left(\frac{ab}{p}\right)$

The Jacobi symbol is an extension of the Legendre symbol for composite $n = pq$:

$$\left(\frac{a}{n}\right) := \left(\frac{a}{p}\right)\left(\frac{a}{q}\right)$$

In a characteristic dexterity as in [27], we expose the computational processes of Legendre (Jacobi) symbol. Let p be an odd prime, we will deduce the algorithm for deciding quadratic residuosity in $Z_p^{\bullet}$.

Proposition 1 (Euler's Criterion): Let p be an odd prime. Then $a \in Z_p^*$ is a quadratic residue modulo p if and on if $a^{(p-1)/2} \equiv 1 (\mathrm{mod}\, p)$.

For the composite $n = pq$, there also exist efficient algorithms to compute the Jacobi symbol of a number, even if the prime factorization of n is not known. However, in contrast to the case Z_p^*, this does not yield a tool for deciding quadratic residuosity in Z_p^*. A quarter of the numbers in Z_p^* are quadratic residues while half of them have Jacobi symbol 1. The numbers having Jacobi symbol 1 while being quadratic non-residues are called pseudo-squares

Definition 4 (Quadratic Residuosity Problem (QRP)): Given a composite integer $n = pq$ and $a \in Z_n^*$ with $\left(\frac{a}{n}\right) = 1$, decide whether or not a quadratic residue modulo $n = pq$ is known, it is easy to solve QRP by computing $\left(\frac{a}{p}\right)$, since a is a pseudosquare if and only if $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. It is these two facts that Goldwasser and Micali based the first semantically secure cryptosystem upon as we shall the role of this cryptosystem as theoretical examples in Section III.

There are no efficient procedures known for solving the Quadratic Residuosity Problem if the factorization of n is unknown. The Quadratic Residuosity assumption states that for sufficiently large two prime numbers p and q, for every real life algorithm, it is infeasible to solve QRP. Nonetheless, if the factorization

III. Chosen Ciphertext Attack Security-Indistiquishability

In [9] and [13] show the practical usage of the public-key encryption scheme that would want the plaintext space to be a set of binary string with length n. That is, $\{0, 1\}^n$ or simply a bit-string $\{0, 1\}^*$ and should be independent of the public key. Haven said and done, this public-key as it stands cannot be applied for homomorphic encryption unless it is padded.

If a system has a cryptosystem that has the property of indistinguishability, an adversary would not be able to find similarity between the pairs of the two different ciphertexts based structurally on the encrypted messages presented to him. For any encrypted text that has indistingushability Chosen Plaintext Attack (IND-CPA). We note in passing that for such a CPA, an adversary has access to a decryption oracle that behaves like a black-box and takes a ciphertext as its input and outputs the corresponding plaintext that security would be nondeterministic. In a nutshell, this implies that every time a bit is encrypted, the ciphertext should be (with absolute probability polynomial time) different such that security may be attained [27]. This requires that encryption parameters should be randomly chosen for the system to remain secure and indistinguishable. The generation of different types of ciphertexts based on the same plaintext is achievable through probability random key generation.

The unconditional declaration of Shannon in 1949 proved that except one-time pad, the security of any other cryptosystem or any encryption scheme can be evaluated. In [26], Shannon introduced the notion of perfect secrecy or unconditional security that characterized encryption scheme for which the knowledge of the ciphertext does not give any information either about the plain text or about the key. With this hypothesize, he proved that one time pad is perfectly secure and any other scheme neither symmetric nor asymmetric has proved unconditionally secure. As far as asymmetric schemes are concerned, their security depends on the hardness of mathematical structure used to design the scheme.

A chosen plaintext attack is secure if for some probabilistic polynomial time (PPT) and a hypothetical adversary A, it holds that:

$$Advert[A] \stackrel{\Delta}{\equiv} | pr[A(pk, evk, HE, Enc(0) = 0) = 0] - pr[A(pk, evk, HE, Enc(1) = 0) = 1$$

where by definition, $(pk, evk, sk) \leftarrow HE.Keygen(1^{\lambda})$

Given a public-key encryption scheme $\Pi = (Gen, Enc, Dec)$ and some adversary who is listening over the experiment for CPA being indistinguishable, can be modeled in this algorithmic pattern:

1. $Gen(1^n)$ is run to obtain keys (pk, sk);
2. The adversary A is given a pk and outputs a pair of messages m_0, m_1 with $|m_0| = |m_1|$, (these messages must be in the plaintext space associated with pk)
3. A random bit $b \leftarrow \{0, 1\}$ is chosen, and then the ciphext $c \leftarrow EnC_{pk}(m_b)$ is computed and given to A. Call c the challenge ciphertext.
4. A output a bit b'; and
5. The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise.

A public key encryption scheme $\Pi = (Gen, Enc, Dec)$ has indistinguishable encryptions in the presence of an eavesdropper if for all probabilistic polynomial-time (PPT), adversary A, there exist a negligible functional defined as:

$$\Pr[PubK_{A,\Pi}^{eav}(n) = 1] \leq \frac{1}{2} + negl(n)$$

where the probability is taken over the random coins used by A, as well as the random coins used to generate (pk, sk), choose b, and encrypt m_b.

IV. Partially Homomorphic Public Key Encryption

Homomorphic encryption Scheme allows some complex mathematical operations to compute on encrypted data without revealing the contents of the original plaintext.

Definition 5; (Homomorphic Encryption): This definition is an extension of the public key encryption scheme (PKE) as promulgated by [24]

A homomorphic encryption scheme is defined as quadruple-tuple $E = (Gen, Enc, Dec, Eval)$ with an extension of two group operators $\langle +, * \rangle$ over the plaintext M and the Ciphertext C, respectively. By definition, the encryption algorithm Evaluate is a "homomorrphism" between the plaintext space M and the ciphertext space C if

$$M_1 + M_2 = Dec(sk, C_1, C_2)$$

where, $C_1 = Enc(pk, M_1, \text{Eval})$; and, for arbitrary M_1 and $M_2 \in M$

A homomorphic encryption model consists of the following four algorithmic parameters as a further clarification:

$$KeyGen \quad (\lambda)$$

A. Implementation of Partially Homomorphic Encryption Scheme

This section gives implementation of partially homomorphic scheme examples as presented in [19]. The section presents [23], [24], [27] and [7].

The inception of this problem has brought along an open problem in cryptography, and most especially in the area of Confidentiality and Integrity security. The first ever system to offer computational succor to this area of study was proposed by [4]. However, encryption systems that operate on only operation had been in existence long after [25] seminar paper

Example 1 (RSA Homomorphism): Consider $n = pq$ where p and q are large primes, meaning that $gcm(p, q) = 1$, that are selected randomly. Select a form $\varphi(n) = (p-1)(q-1)$ ∋ $ab = 1 \bmod \varphi(n)$, implying that $a = b^{-1} \bmod \varphi(n)$. The parameters (n,b) are public while the three parameters p, q *and* a are private. Compute the encryptions:

$$En_k(x) = x^b \bmod n = y$$

where x is the plaintext and y is the ciphertext computed;

$$Dec_k(x) = y^b \bmod n = x$$

The homomorphism of the RSA is computed in simplified steps as follow:

Consider two plaintexts x_1 *and* x_2. Then their Homomorphism is expressible as:

$$\begin{aligned} En_k(x_1)En_k(x_2) &= x_1^b x_2^b \bmod n \\ &= (x_1 x_2)^b \bmod n \\ &= En_k(x_1 * x_2) \end{aligned}$$

Example 2 (El Gamal): Randomly select a large odd number and a generator $\alpha \in Z_p^*$, select a *and* β such that

$$\beta = \alpha^a \pmod p$$

Make p, α *and* β public; the private key. Provide a parameter $r \in Z_{p-1}^*$ also a secret random number. Then,

$$En_k(x, r) = (\alpha^r \bmod p, x\beta^r \bmod p)$$

The homomorphism of El Gamal Homomorphic Cryptosystem can be computed theoretically from its cryptosystem as follow:

For two arbitrarily plaintexts x_1 *and* x_2 and compute:

$$\begin{aligned} &En_k(x_1, r)En_k(x_2, r) = \\ &(\alpha^{r_1} \bmod p, x_1\beta^{r_1} \bmod p)(\alpha^{r_2} \bmod p,\ x_2\beta^{r_2} \bmod p) \\ &= (\alpha^{r_1}\alpha^{r_2} \bmod p, x_1, \beta^{r_1}\beta^{r_2} \bmod p) \\ &= (\alpha^{r_1+r_2} \bmod p,, \ (x_1 + x_2)\beta^{r_1+r_2} \bmod p \\ &= En_k(x_1 + x_2 . r_1 + r_2) \end{aligned}$$

The usefulness of El Gamal additive Homomorphism to the Big Data Analytics:

E-cash and e-voting will benefit from the additive homomorphism in the cloud computing environment. To achieve this aim, however, then algorithm will look theoretically in this abstractive formation:

$$En_k(x, r) = (\alpha^{r_1} \bmod p, \alpha^x \beta^r \bmod p)$$

The generic Additive El Gamal Homomorphic Cryptosystem will compute uniquely to:

$$\begin{aligned} &En_k(x_1, r_1)En_k(x_2, r_2) = \\ &(\alpha^{r_1}\beta^{r_1} \bmod p, x_1\alpha^{r_1} \bmod p) * \\ &(\alpha^{r_2} \bmod p,\ \alpha^{x_{x_2}}\ \beta^{r_2} \bmod p) \\ &= (\alpha^{r_1}\alpha^{r_2} \bmod p, x_1, \alpha^{r_1}\beta^{r_2} \bmod p) \\ &= (\alpha^{r_1+r_2} \bmod p,, \ \alpha^{(x_1+x_2)}\beta^{r_1+r_2} \bmod p \\ &\quad = En_k(x_1 + x_2, r_1 + r_2) \end{aligned}$$

The modified homomorphism is that $Dec_k = \alpha^x$, introduces the discrete logarithm problem into the Somewhat Homomorphic Encryption into the decryption scheme. For large enough ciphertexts, this will become impractical.

Due to the complication of the El Gamal Cryptosystem, let us envision further another alternative that takes into account the additive property of exponentiation; which can operate with extra decryption time cryptosystem as outlined in [12].

Example 3 (Paillier Homomorphism): First, let us face some preambles as construed in [12] and presented as an asymmetric PKE (Public Key Cryptosystem) that could be applied on polynomial probabilistic time (PPT) Somewhat Homomorphic Encryption Scheme that is based on multiplication axiom. This polynomial composite residuosity assumption runs as follows:

Select a composite number that is an integer; it is hard to determine if y exists such that $Z = y^n (\mathrm{mod} n^2)$

This work was further extended in [28], a Homomorphism we will not discuss in this paper.

Now the race to Paillier runs in these sequential steps:

Select randomly two large primes $p \; and \; q$ and compute $n = pq$

Suppose λ denotes the Carmichael function:

$$\lambda(n) = lcm(p-1, q-1)$$

Select a random number $g \in Z^*_{n^2}$ ∋ $L(g^{\lambda} \bmod n^2)$ is invertible modulo n where $(L(u) = \frac{u-1}{n})$ and g are public key; $p \; and \; q \, (or \; \lambda)$ are private entities. For a given plaintext x and resulting y ciphertext, select a random $r \in Z^*_n$ Then we have the ciphertext:

$$En_k(p, r) = g^m r^n \bmod n^2 = c$$

And the decrypted computed to obtain the plaintext in this manner:

$$Dec_k(y) = \frac{L(y^{\lambda} \bmod n^2)}{L(g^{\lambda} \bmod n^2)} \bmod n$$

So far, we have been studying the Somewhat Homomorphism Encryption Scheme. We will now look into Fully Homomorphism Encryption (FHE) Scheme in the next Section

Example 4 (Goldwasser-Micali Homomorphic Encryption Scheme): This theoretical example is from [7] description of homomorphic scheme. This brand of cryptosystem is additively homomorphic and it is based on the quadratic residuosity assumption. In other words it is hard to distinguish quadratic residues with Jacobi symbol (as we theorized in Section II above). The Godwasser-Micali cryptosystem works in this pattern:

1. Generate two keys; public key pk and secret key Sk. Based on the Gen algorithm, two large primes p and q are chosen such that $N = pq$. Gen chooses a random number $y \in Z^*_N$ that is non-square modulop and non-square modulo q. Gen returns the public key $pk = (N, y)$ and the private key $sk = (p.q)$
2. For the encryption algorithm Enc, of the bit say, $b \in \{0, 1\}$: The algorithm will choose a random integer r and the public key will be (N,y) to obtain the ciphertext $c \leftarrow r^2 * y^b (\mathrm{mod} N)$. The random choice automatically makes the output values a probabilistic polynomial time value; thus a nondeterministic ciphertext.
3. The Decryption algorithm Dec takes a ciphertext c and private key sk=(p,q) as input, and outputs the message b, where b=0 if c is quadratic residue, otherwise, m=1

Now, if the quadratic residuosity problem is intractable, then the Goldwasser-Micali scheme is semantically secure. In other words, an adversarial A has negligible advantage in the following Game of indistinguishability chosen ciphertext attack-IND-CPA

$$\begin{vmatrix} (sk, pk) \leftarrow K(1^{\lambda}) \\ (m_0, m_1) \leftarrow A(pk) \\ \quad c \leftarrow A(Enc(m_{\beta}, pk), \beta \leftarrow \{0, 1\} \\ \quad \beta' \leftarrow A(m_o, m_1, c.pk) \\ return \; \beta' \end{vmatrix}$$

At the end of this game, the attacker's advantage

$$Adv^{IND-CPA}_{Enc} = \begin{vmatrix} pr\left| Exp^{IND-CPA}_{ENc,A,} = 1 \right| \beta = 1 | \\ -\left| pr\left| Exp^{IND-CPA}_{ENc,A,} = 1 \right| \beta = 0 \right| \end{vmatrix}$$

The Homomorphic encryption property could be processed as follow: for any $b, b' \in \{0, 1\}$ the following equefinition is clearly in order by definition:

$$Dec(Enc(b, pk) x Enc(m', pk) sk) = m \oplus m'$$

In homomorphic encryption scheme, the encryption algorithm encrypts one bit at a time, hence, in order to encrypt a binary string, we need to encrypt every bit individually. Since this paper is not purely a cryptographic exposition, further details will be out of scope.

V. Fully Homomorphic Encryption Scheme

The authors were the progenitors of the idea of homomorphic cryptosystem in [13], [14]. The paper vented the idea of Privacy Homomorphisms and outlined four fundamental possible encryption functions which included the RSA as additive and multiplicative privacy homomorphism. With the advent of this rich foresighted paper, many encryption models have been developed, but they were either used as addition or multiplication homomorphic computations. Clear examples of these are [8], [10], [12], [25], [26] and [28]. These were known as Partial Homomorphic schemes because they can only operate on one binary operation at a time; that is none of the scheme could support both addition and multiplication computations simultaneously in unique computational process. In a more detail observations of Fully Homomorphic Encryption, [1] developed an encryption scheme that was able to perform many additions and one single multiplication on ciphertexts. In June [4] and from IBM

implemented the first fully homomorphic encryption scheme that was able to to perform many additions and multiplications using ideal lattices and bootstrapping technique. They laid the foundation of the recent FHE schemes including [8] and [29].

The cardinal target of Fully Homomorphic Encryption is to enhance the security of cloud computing. The data is first encrypted and sent to the cloud computing environment. In the cloud, the sent encrypted data is subjected to computations using some functions without transforming the encrypted data into plaintext. When the computed data is queried, the encrypted output is sent back for decryption; using the privatekey, the plaintext is recovered from the computational out function.

A. Properties of Fully Homomorphism Encryption

The computational processes that is carried out is based on the fully Homomorphic scheme has no limited number of manipulations that can be executed on it. The computational processes are usually performed on circuit gates combinations of AND gates and XOR gates whose inputs are ciphertexts $C_1, \ldots, C_t$. Besides, the triple-tuple algorithms required in any public key cryptosystem $(KeyGen, Encryption, Decryption)$, a homomorphic encryption scheme demands that there should be another algorithm in the cloud known as the Evaluate Algorithm. The Evaluate Algorithm takes as input the public key pk, a circuit function f and a tuple of ciphertexts $C =< C_1, \ldots, C_t >$ which are used as input to f Fully Homomorphic Encryption scheme can query a search engine, without revealing what is being searched for (in this context, the engine does the computations on encryptions of the information that it does not know). Put more succinctly, FHE in its simplified application has the following unified property. Consider a ciphertext C_i that decrypts to the plaintext M_i, put formally:

$$Decrypt(C_i) = M_i$$

where by definitions $M_1's$ and $C_1's$ are some elements of some polynomial ring (a ring consists of two operations, addition and multiplication). The homomorphic encryption scheme is given by the following two models $Dec_{sk}(C_1 + C_2) = m_1 + m_2$ and $Dec_{sk}(C_1 * C_2) = m_1 * m_2$

In other words, decryption is homomorphic with respect to the two operations; addition and multiplication. Being fully homomorphic means that whenever f circuit functions that is composed finitely will compute any additions and multiplications in the ring, that is:

$$Decrpt(f(C_1,\ldots,C_t)) = f(m_1, \ldots, M_t)$$

If the cloud (which we visualize as the adversary) can compute efficiently $f(C_1,\ldots,C_t)$ from the ciphertexts $C_1, \ldots, C_t$ without learning any information about the corresponding plaintexts $m_1, \ldots, m_t$, then the scheme is efficient and secure.

Another property of FHE is Ciphertext Compactness, This requirement postulates that for FHE to decrypt fully, there the polynomial function should not increase in size. Hence the Ciphertext should maintain a low polynomial degree or be bounded above. In other words, that the ciphertext should remain bounded or compact, independent of the function, f this is known as compact ciphertexts requirement in the parlance.

Another requirement is that FHE schemes can be either public key (where the encryptor or and decryptor share a key that is used for both encryption and decryption).

It is generally known that all ciphertexts contain some noise or error in them. This noise grows as homomorphic encryption computation is being manipulated on the ciphertext. Thus makes the ciphertext to increase in size as the polynomial degree grows. With this phenomenal tendency, the decryption of the computed ciphertext becomes infeasible due to the noise impediment. Besides, this growth in ciphertext violates the principle of ciphertext compactness.

B. Gentry's Fully Homomorphic Encryption Algorithm

In this subsection, we discuss the format for the construction of Gentry's FHE [4].

Gentry's homomorphic model consists of three steps:

1. Somewhat Homomorphic;
2. Quashing; and
3. Boostrapping

These three algorithms support a limited number of additions and multiplication on the ciphertexts [2], [4] and [9]. In Gentry initial computational processes; start with the somewhat homomorphic encryption (SHE) and then Bootstrapping. Bootstrapping leads to the formation of fully homomrphic scheme as we see below in this section. Somewhat Homomorphic Encryption supports a limited number of additions and multiplication on ciphertexts. This reduction in computational extension of homomorphism is caused by the inherent error or noise as the computation processes. Once the noise component reaches a certain threshold, say k, the computational encrypted ciphertext does not decrypt correctly further. For instance, suppose the noise is N and the threshold is k, then, the bound is reached after only $\log_2 k$ levels of multiplication. In SHE schemes, the ciphertexts could get larger that could lead to message expansion; this violate the compactness of ciphertext requirements.

If one wants to achieve FHE, this inherent computational enlarged noise must be solved. Gentry's solution is to decrypt the ciphertext, but homomorphically and have it sent to the cloud with the encrypted message. Such a private a key should be able to decrypt itself once in the cloud so as to decrypt the encrypt message at some phases.

Reference [4] proposed the first Fully Homomorphic Encryption scheme. The description of the scheme:

This is centred on a function which introduces a certain level of noise into the encryption scheme that generates the following compounded issues:

1. Each operation on the ciphertext results in compounding the inherent noise;
2. Resolved this problem by bootstrapping the encryption;
a. Each re-encryption cuts down the noise;
3. The operation of FHE is based on Ideal Lattices: that is Ideal in number theory which
i) allows for new complex circuit implementation;
ii) corresponds to the structure of ring homomorphism

The FHE as outlined [4] includes Somewhat Homomorphic scheme as part of its execution operation. The intention is to disallow the algorithm to converge locally. Let us idealize concretely the bootstrapping noise algorithm for the SWHE.

1. KEYGEN Evaluation: Output a random odd integer p;
2. For bit $m \in \{0,1\}$, let $m' = m \bmod n$; where m' is even if m=0, odd if m=1. Select a random q.

$$ENCRYPT_E(m, p) = C = m^1 + pq$$

m' is the noise associated with the plaintext

3. Let $C' = C \bmod p \ and \ C' \in (-\frac{p}{2}, \frac{p}{2})$

then, $DECRYPT(p, c) = c' \bmod 2$

The noise encrypted C' is considered to be the noise associated with the ciphertext (that is, the shortest distance to a multiple of p)

The Homomorphism; Multiplication:

Let $m_1, m_2 \in \{0,1\}$ then,

$$Enc_{\in}(m_1, p)Enc_{\in}(m_2, p) = (m_1' + pq_1) * (m_2' + pq_2)$$

The decryption:

$$Dec(c) = (m_1' + pq_1)(m_1' + pq_2) \bmod p \bmod 2$$
$$= m_1'.m_2' \bmod 2 = m_1.m_2$$

The compounding noise (m_1', m_2') stems from the loss of some homomorphic property after a certain number of operations. To disallow the computation to stop it's computational processes due to the inherent noise, [4] introduced bootstrapping into the FHE. This allows the process to take in more inputs and processes for a longer period and therefore improves performance. Let us review bootstrapping theoretically without any mathematical pronouncement:

Gentry in his PhD thesis, [4] introduced the working idea for bootstrapping. He opined that to go bootstrapping is to go from somewhat Homomorphic encryption to Fully Homomorphic encryption. He gave a vivid illustration on this that enables one to go through bootstrapping. One can include as a part of the public key, an encrypted private key. The idea is this, when the noise gets too large during computation in a remote server in the cloud, the encryptor can then use the somewhat homomorphic encryption scheme to evaluate the decryption function that is applied to the ciphertext using the encrypted key. This self-re-encryption process produces a new ciphertext that is compact and less noisy. However, for this sequential method to work, it is necessary for the Somewhat Homomorphic Scheme to be "circular secure". Circular secure implies that it must be able to re-encrypt itself. This is themean reason that motivated him into introducing the Ideal Lattice into the Fully Homomorphic Scheme.

In a final stepping down, bootstrapping of the Somewhat Homomorphic Cryptosystem allows for the reduction of the noise and therefore makes the ciphertext more compact and less noisy which allows no limit of operations. Nonetheless, the combination of the noise production that follows the noise reduction completely makes the scheme terribly impractical. Ever since the observation of this bootstrapability, more schemes have been introduced (that we cannot at this level give instances due to space constraints) to try and decrease this complexity, but all depend largely on the bootstrapping idea.

Despite all the impracticability, [4], [6] made much break through to Cryptography with his bootstrapping Somewhat Homomorphic Encryption Scheme that strengthens the implementation of the Fully Homomorphic Cryptosystem Scheme. With this onerous contribution and its security attendance, the security of Big Data Analytics has been boosted in the Cloud computing environment.

VI. Application of Fully Homomorphic Scheme in the Cloud Computing

The section makes some insightful discussion on the implementation of FHE in the Cloud Computing environment. The discussion is centred on the application of the database server and client implementing Homomorphic Encryption architecture as abstracted in [15], as shown in Fig. 1. Fig. 1 describes the computational process of the fully Homomrphic Encryption scheme (FHE)

In the implementation, analytical performance of the FHE cryptosystem works on the virtual platform as a cloud server. A VPN acts as duct channel for the transmission of the encrypted data to the virtual server which is the storing caveat of the encrypted data. It is also the platform on which the fully homomorphic encryption processes are computed as outlined in Section IV and demonstrated in Fig. 1.

The Authors in [15] abstracted the Database server communicating with the client using the FHE cryptosystem.

In a similar abstraction, [15] abstracted the scenario that is illustrated in Fig. 2.

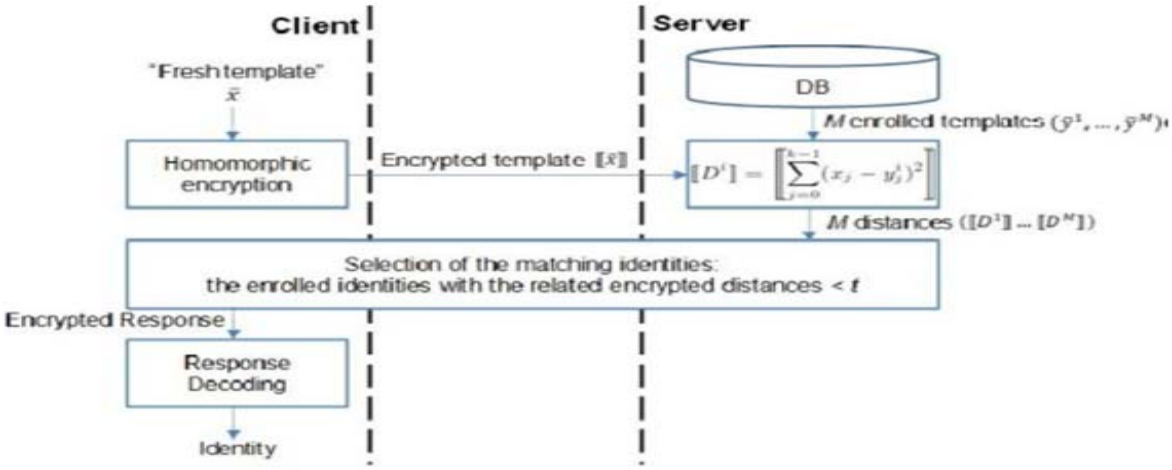

Fig. 1 Database-server & client implementing homomorphic encryption; courtesy of [15]

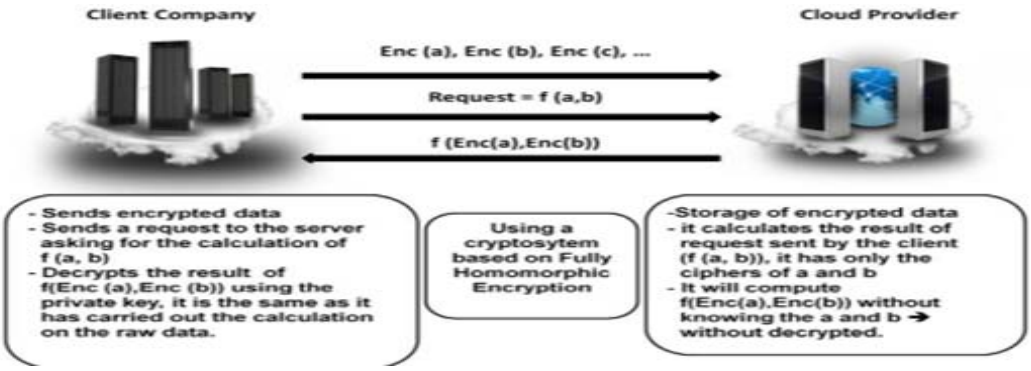

Fig. 2 Implementation of FHE in the cloud between the server and the client: courtesy of [15]

VII. Conclusion and Perception of Future Works

The research theoretic has been targeted at the computational outlooks of the Fully Homomorphic Encryption scheme and as such some number theory and algebra have been useful in the computational application of Big Data Analytics in the Cloud. The acknowledged problem with Cloud Computing is the issue of privacy and confidentiality of both the Client and the computation of the data stored in the Cloud Computing. The solution to the problems was uniformly solved by sending the data encrypted to the Cloud. We must explicitly make the assertion that other mechanisms exist for secure computation

in the cloud, they generally require the different data providers to exchange information such as the private key. FHE schemes are public key schemes; therefore, they are much better suited for the scenario where many sources of data coexist.

However, this needs to have supporting and useful computations on the encrypted data. This we could achieve through the application of Fully Homomorphic Encryption Scheme as developed by [4], [5]. FHE is a way of supporting such computations on encrypted data Gentry's encryption scheme still has the squashing and bootstrapable problems and would want to seek much more to achieve more plausible operational efficiency

References

[1] Z. Brakerski, C. Gentry, And V. Vaikuntanathan,"(Leveled) Fully Homomorphic Encryption Without Bootstrapping," In Itcs, 2012
[2] D. Boneh, E. J In Goh, And K. Nissim, "Evaluating 2-Dnf Formulas On Ciphertexts. In Theory Of Cryptography"; Conference, Tcc 2005, Volume 3378 of Lecture Notes in Computer Science, Pages, 325-341, Springer, 2005
[3] W. Diffie, M. E Hellman, "New Directions in Cryptography, IEEE Transactions on Information Theory, Volume IT-22, No. 6, (Online) http://ee.standford,edu/~hellman/publications/24.pdf
[4] C. Gentry; "A Fully Homomorphic Encryption Scheme" http://crypto.standford.edu/craig.thesis.pdf
[5] C. Gentry and S. Halevi, Implementing Gentry's Fully Homomorphic Encryption Scheme, Preliminary Report, (Online). 2009 http://reasearcher,watson.ibm.com/researcher/files/usshail/fhe-implementation.pdf
[6] C. Gentry, "Computing Arbitray Functions Data" Communication of the ACM, Vol. 53, pp 97-105, 2010
[7] S. Goldwasser and S. Micali: "Probabilistic Encryption & how to play mental porker keeping secret all partial information", In proceedings of the 14th ACM symposium on the Theory of Computing (STOC '82), New York, NY, USA, pp,21-53
[8] Goldwasser, S. and S. Micali, " Probabilistic encryption", In Journal of Computer and System Sciences, 1984, Vol 28, no. 2, pp270-299
[9] J. Katz & L. Yehud, "Introduction to Modern Cryptography", Chapman& Hall/Crc Cryptography and Network Security, 2007.
[10] K. McCaney, "New Encryption method with promises end-to-end cloud security" http://gcn.com/Articles/2013/06/Encryption.end.to.end.cloud.security.aspx?Page =1
[11] J. Nittin, Saibal K. Pa and Dhananjay, K. Upadhyay; "Implementation and Analysis of Homomorphic Encryption Schemes, International Journal on

[12] P. Paillier: Public-key cryptosystems based on composite degree residuosity classes. In 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic, Volume 1592,1999
[13] P. Meli and Timothy Grance, The NIST Definition of Cloud Computing, Special Publication, 2011 800-145, Recommendation of the National Institute of Standards and Technology.
[14] L. Ronald, Rivest, L. Addleman, and M. L. Dertouzos; "On Data Banks and Privacy Homomorphism, Chapter on ata Banks and Privacy Homomorphisms, pages 169-180, academic press 1978
[15] S. Bajpai and P. Srivastava, "A Fully Encryption Implementation Cloud Computing, International Journal of Information and Computation Technology", ISSN 09742239 Volume 4 (2014), pp. 811-816; copyright International Research Publications House. http://www.irphouse.com
[16] Y. Sophia, G. Vijay. S. Nabil, S. Emily, and Y. Arkady, "A Survey of CryptographyApproaches to Securing Big Data Analytics in the Cloud", MIT Lincoln Laboratory, availalable (online), 2014: http://www.ieee.hpcc.org/2014/CD/index_htm_files/FinalPapers/28.pdf
[17] V. Dijk, M. Gentry, C. Halevi, V. Vaikuntanathan: "Fully Homomorphic Encryption over the Integers." Advances in Cryptology-EUROCRYPT2010; 24-42
[18] V. Vaikunthanathan, "Computing Blindfolded: New Developments in Fully Homomorphic Encryption," Supported by NSERC Discovery Grant and By DARPA under Agreement number FAS8750-11-1-0225. University of Toronto, 2013
[19] V. O. Waziri, J. K. Alhassan, O. Morufu and I. Ismaila, "Big Data Analytics and the Epitome of Fully Homomrphic Encryption Scheme for Cloud Computing Security", International Journal of Developments in Big Data and Analytics Voolume 1 No.1, 2014, pp 19-40
[20] White Paper Big Data Analytics SAS www.sas.com/.../big-data-meets-big-data-analytics-10577.pdf
[21] White Paper; An Oracle while Paper (March, "Advanced Analytics in Oracle Database" www.revolution-nalytics.com
[22] N. Rouda, Senior Analyst, White Paper, "Getting Real About Big Data: Build Versus Buy. Enterprise Strategy Group", www.narius.com/esg-brief-on-big-data-analytics/esg-geetting-real-bigdata-2228170.pdf
[23] N. Rouda, Senior Analyst, "ESG White Paper, Getting Real About Big Data: Build Versus Buy. Enterprise Strategy Group", www.narius.com/ esg-brief-on-big-data-analytics/esg-geetting-real-bigdata-2228170.pdf
[24] R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", IEEE Trans. Inform. Theory, 1978
[25] T. El Gamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions On Information Theory, Vol. It-31, No. 4, July 1985
[26] C. E. Shannon and Warren Weaver, "The Mathematical Theory of Communication", University of Illinois Press, 1949
[27] I. Damgard, M. Jurik and J. B. Nielsen, "A Generalization of Paillier's Public-Key System with Applications to Electronic Voting", Aarhus University, Dept. of Computer Science, BRICS, 2001
[28] J. D. Cohen and M. J. Fischer, "A Robust and Verifiable Cryptographically Secure Election Scheme (Extended Abstract)", YALEU/DCS/TR-416 July, 1985
[29] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic, 2012