Advanced Cryptographic Techniques for Secured Cloud Computing

Nilotpal Chakraborty
M.Tech Student, School of Future Studies and Planning
Devi Ahilya University, Indore, India

G K Patra
Principal Scientist, CSIR- Fourth Paradigm Institute
Council of Scientific and Industrial Research
NAL Belur Campus, Bangalore, India

ABSTRACT
The past decade of computing has witnessed a number of new computational models and the most prominent among them is Cloud Computing. Cloud Computing is a paradigm shift that helps a user with internet based computing services that can be accessed from anywhere on any platform. But despite of its advantages, it is yet to gain total trust from users, the primary reason being its security issues. Though some standard organizations have developed a number of security compliance guidelines that need to be followed to ensure security and quality of services in the cloud, security assurance in real terms remains to be undercover. This paper discusses about the two most promising cryptographic techniques, that are, if implemented correctly can effectively mitigate the security threats and can help in an increased uses of cloud computing.

General Terms
Security, Cryptography

Keywords
Cloud Computing, Fully Homomorphic Encryption, Functional Encryption

1. INTRODUCTION
Cloud computing has been one of the buzz word during the last couple of years in the field of computing. It is a computing model that delivers IT resources to its users via Internet and therefore, sometimes called as Internet Computing. The formal definition of cloud computing is described in [1] as—

"It is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction".

Cloud computing offers various services such as data, applications, storage, servers etc to the users with the help of its service models viz. Software-as-a-service, Platform-as-a-service, Infrastructure-as-a-service. It offers a ubiquitous, scalable, multi tenant IT services to the users over the internet. Users can have all their data stored in the cloud which they can access from anywhere from an internet enabled device. The services are generally provided by some vendors, known as Cloud Service Providers (CSP) and they charge for them based the usage. Thus users do not need to install the system on their premises and can use the same as provided by the CSPs. This frees them from maintaining the system on their own and can still have the same computational power.

But as with the traditional mode of computing, security issues bother the cloud computing model also and which is the primary reason that hinders the wide spread use of it. The data and information stored on the cloud can be highly valuable to individuals with malicious intents. Due to the flexibility and convenient services cloud provides, a lot of personal information and potentially secure data is now being transferred to the cloud. This makes it critical to understand the security measures that the CSPs has in place, and it is also very important to ensure personal precautions and proper measurement to assess the security of the data [2]. There are numerous security issues for cloud computing as it encompasses many technologies including computer networks, databases, data analytical tools, virtualization, resource sharing and scheduling, transaction management, load balancing, concurrency control and memory management [3]. Thus, security challenges and concerns for many of these systems and technologies are applicable to cloud computing.

The best way to address the security issues is to encrypt all the data and then store it into the cloud. But with traditional encryption schemes, once the data is encrypted, it is completely meaningless unless decrypted. Moreover, traditional ciphers encrypt All-or-Nothing which creates a problem in leveraging the total power of cloud computing services. Thus it is need to devise some advanced mode of encryption techniques that can provide encryption of data in one hand, and should also provide the capability to manipulate those data over the cloud. Two of such advanced cryptographic schemes are Fully Homomorphic Encryption and Functional Encryption, which are going to be discussed primarily in this paper.

The outline of the paper is organized as follows— Section 2 discusses about fully homomorphic encryption, section 3 talks about functional encryption. Section 4 talks about the implementation of fully homomorphic encryption and functional encryption on cloud computing. Section 5 and section 6 respectively discusses about our contribution towards the field of functional and homomorphic encryption and its related works.

2. FULLY HOMOMORPHIC ENCRYPTION
Fully Homomorphic encryption (FHE) has been an intense area of research for the past three decades, was originally introduced by Rivest, Adleman and Dertouzos in 1978 [4], soon after the development of RSA encryption scheme. Rivest et.al named it Privacy Homomorphism and showed that the Basic RSA [5] is a multiplicatively homomorphic encryption scheme -i.e, given a RSA public key $pk= (N,e)$ and cipher texts $C_i= M_i^e \ mod \ N$, one can efficiently compute $\Pi_i C_i=(\Pi_i M_i)^e \ mod \ N$, which is eventually a cipher text that encrypts the product of the original plaintexts. With this invent, they asked a basic question: What can

one do with a scheme that is fully homomorphic. The answer to this question is that one can compute arbitrary number of computations on the encrypted data without the need to decrypt it anywhere. The following figures depicts the functioning of FHE—

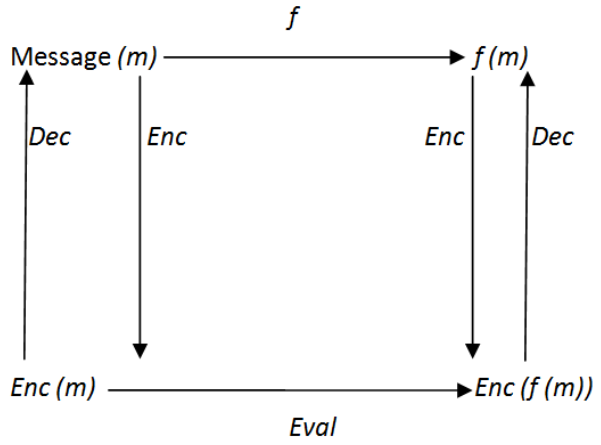

Figure 1: Fully Homomorphic Encryption

For the past three decades, Scientists and Researchers tried immensely to develop an encryption scheme that supports arbitrary number of computations on the cipher texts, thus enabling fully homomorphic encryption. Despite of their hard work and efforts, most of the cryptographic schemes are turned out to be partial homomorphic, such as RSA scheme, El-Gamal cryptosystem [6], Paillier cryptosystem [7] etc. supporting either multiplication or addition of cipher texts. But in 2009, IBM researcher Craig Gentry in his breakthrough work showed the first plausible fully homomorphic encryption scheme based on the ideal lattices [8].

Any public-key encryption scheme has basic three algorithms. KeyGen: Generates a key/ a pair of keys, Encrypt: Takes the plain text and the key and produces a cipher text, Decrypt: Takes the secret key and the cipher text to produce the original plain text. For a scheme to be Homomorphic, there is another algorithm called Eval. The Eval takes the set of cipher texts and set of public keys to perform some specific operations on them.

The construction of Gentry's Fully Homomorphic Encryption scheme [9] is as follows:

- *KeyGen(λ)*: The secret key is an odd η-bit integer: $p \longleftarrow (2Z + 1) \cap [2\eta-1, 2\eta)$.

For the public key, sample $x_i = pq_i + r_i$, for $i = 0, \ldots, \tau$. Reconstruct so that x_0 is the largest. Restart unless x_0 is odd. The public key is $pk = x_0, x_1, \ldots, x_\tau$.

- *Encrypt (pk, m∈{0, 1}):* Choose a random subset $S \subseteq \{1, 2, \ldots, \tau\}$ and a random integer r in $(-2\rho, 2\rho)$, and output $c=(m + 2r + 2 x_i) \bmod x_0$.
- *Evaluate (pk, C, ci):* Given the binary circuit C with t inputs, and t cipher texts c, apply the addition and multiplication operations of C to the cipher texts, performing all the operations over the integers, and return the resulting integer.
- *Decrypt (sk, c):* Output $m=(c \bmod p) \bmod 2$.

The above scheme to work correctly, r needs to be sufficiently small as compared to the primary number p. But as the operations are carried out on the cipher texts, the noise associated with the cipher texts grows and after a certain threshold value, it becomes impossible to decrypt it to its original plain texts. This particular noise problem reduces the number of allowable homomorphic operations that can be performed.

3. FUNCTIONAL ENCRYPTION

Functional encryption is quite a new in cryptography and is an asymmetric key algorithm that allows the secret key to decrypt only some specific functions of the original texts, without revealing any other information. This type of encryption scheme posses a great implication as with the increase of distributed computing and cloud computing. Functional encryption was introduced in 2005 by Amit Sahai and Brent Waters [10] which supported the evaluation of some specific functionality. By 2012, several researchers around the world have developed Functional Encryption schemes that support arbitrary functions.

The notion of functional encryption has been developed from the existing cryptographic techniques like Fully Homomorphic Encryption, Yao's Garbled circuit, Identity based encryption (IBE) and Attribute based encryption (ABE). Eventually, IBE and ABE can be termed as special cases of functional encryption that support encryption/decryption of cipher data based on some identity and attribute respectively. Thus functional encryption is an advanced encryption scheme that supports restricted permission based encryption and decryption. The pictorial representation of functional encryption can be depicted as follows—

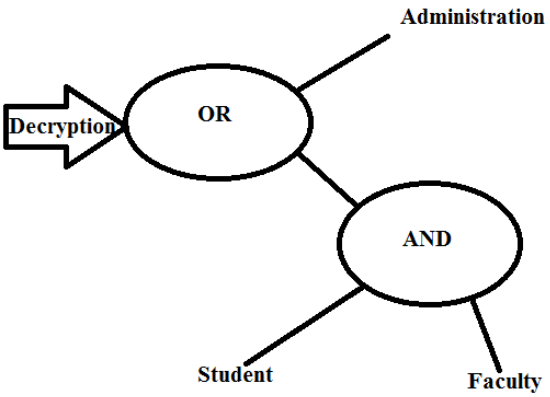

Figure 2: Functional Encryption

The construction of functional encryption consists of the following algorithms:

- *(pk, msk)* ~~←~~ *SetUp(1λ)*: creates a public key *pk* and a master secret key *msk*.
- *sk* ~~←~~ *KeyGen(msk,k):* uses the master secret key to generate a new secret key *sk* for value *k*.
- *C* ← *Encrypt (pk,m)*: uses the public key *pk* to encrypt a message *m*.
- *F(k,m)* ← *Dec(sk,c)*: uses secret key *sk* to calculate a function of the value *C* encrypts.

4. FHE AND FUNCTIONAL ENCRYPTION ON CLOUD COMPUTING

Fully homomorphic encryption (FHE) is the sole way to achieve computability in an encrypted domain. With the help of this, any user can encrypt his/ her data before storing it to the remote server and then perform valuable computations on the encrypted data without the need to decipher it anywhere [11]. This concept, if implemented efficiently, can address the security issues and concerns of Cloud Service providers and also will help in building trust among the Cloud service users. As their data is completely encrypted under a secret key, only known to them, it is ensured that the cloud does not learn anything from their data. Moreover, with the homomorphism, the cloud can compute on the data without the need to decrypt it. FHE can help in building trust among the cloud users with the fact that only the owner of the data and application can actually decrypt it and none else gain anything from it.

Functional encryption, on another hand, supports the group secrecy mechanism by only allowing a portion of the functionality of the cipher text to be revealed. The secret key of the scheme can only reveals specific functionality that restricts the user to learn anything beyond their privilege. For example, suppose for a medical study, students need to conduct a survey about how many patients were infected by swine flu during the period of April, 2012 to March 2013. Now here if we consider that the medical data of the patients are encrypted, then according to the needs of the study, only the number of the patients should be revealed, without deciphering any other information such as patient name, address or phone number. Thus Functional encryption helps in a number of ways where a set of cipher text, though can be seen by everyone, but only a specific portion of it can be decrypted.

5. PERFORMANCE ANALYSIS AND ENHANCEMENT

The inefficiency of Gentry's FHE scheme is the fundamental reason why FHE is yet to be implemented in real life applications. As it works on message space bits, it takes quite a bit of time to actually encrypt or decrypt a long message. Moreover to maintain semantic security of the scheme, it is recommended to use large prime numbers which results in long keys. The initial implementation of the scheme resulted in a key of 2.3 GB and 36 hours to evaluate AES algorithm. Our primary focus towards this field is to reduce this inefficient complexity behind the implementation of FHE and to optimize it for the real life applications, especially on cloud. As part of this study, the authors have analyzed the mathematical background of fully homomorphic encryption and have worked on to understand the reason behind its functionality and inefficiency. Authors have implemented a variant of Gentry's scheme as proposed by VanDijk et. al [12] which is based on approximate GCD based problem. The approximate GCD problem states that given a multiple of any prime number, plus some random number, it is difficult to obtain the original prime number. Having implemented that, the homomorphic property of cipher texts has been observed. Authors have also performed multiplication and addition of cipher texts and then decrypted them so as to obtain the result as if the operation was originally performed on plain texts.

The authors have also implemented the FHE scheme proposed by ZHANG-Tong et.al. [13] in which FHE can be implemented on real numbers. The scheme supports addition, multiplication, subtraction and division of cipher texts. The primary drawback that was observed in this scheme was that, all the operations have to be known before actually performing them. That makes the scheme ineffective in terms of real life scenario, where a user cannot be expected to know in prior what operations he may later perform on his data. Nevertheless, this paper helps us in developing a FHE scheme that can support not only ring operations, but also field operations. If the cipher text space can be represented by a field, then it can be possible to apply all the numerical operations to be performed directly on the user data. That scheme would be much more efficient and effective in real use.

The authors are also working on Functional encryption and following the scheme proposed by Goldwasser et.al. [14] to implement it. In functional encryption, there is a master key that generates public keys and a corresponding secret key based on the specified functionality. Based on the generated secret key, the user can only decrypt that functionality for which this secret key was generated. The security of functional encryption solely depends on securing the master secret key. The focus has been to implement functional encryption to secure cloud computing infrastructure. With the help of open source cloud development technologies, a private cloud has been developed and currently work is being carried out to secure encrypt/decrypt data with the help of functional encryption for restricted data access mechanism.

All our implementations are being done on Linux based computers in C programming language. For handling larger data, GNU Multi Precision (GMP) Library is used. GMP is an open source freely available C/C++ library that helps in handling with larger numbers efficiently.

6. RELATED WORKS

Though Gentry's scheme was not found to be efficient enough to implement, it was undoubtedly a breakthrough in cryptography. Following Gentry's strategy, In 2010, Van Dijk, Gentry, Halevi, and Vaikuntanathan came up with a fully homomorphic scheme (known as DGHV scheme) over the integers, with a reduced public key size of 10.3MB and a cipher text refresh procedure of 11 minutes. Later on in 2011, Brakerski, Gentry and Vaikuntanathan (BGV) devised a scheme based on learning with error (LWE) or ring-LWE (RLWE) problems [15]. It allows one to encrypt vectors of plain text in a single cipher text and to perform any permutation on the underlying plaintext vector while manipulating only the cipher text. Smart and Verceuteren presented a gully homomorphic scheme which has both relatively small key and cipher text [16].

In recent times, Coron et. al [17] proposed a technique to reduce the size of the key of the Van Dijk et. al scheme. In April 2013, HElib (an open source library) was released, via GitHub, that implements BGV scheme, along with many other optimizations to make homomorphic evaluations running faster and much more efficiently.

7. CONCLUSION

Cloud computing offers a great deal of computational benefits to the users by providing multi tenant, ubiquitous, scalable and reliable IT services. While the benefits of cloud computing are clear, it imposes new security issues and challenges since cloud operators are expected to manipulate client data without necessarily being fully trusted. Cryptography provides complete data security by encrypting all the data which is meaningless without its decryption. But if

traditional ciphers are used in securing cloud infrastructure, the primary benefits of it cannot be leveraged. For this, we need advanced encrypting techniques that despite of the data being encrypted, helps in performing operations on it. Two of the most advanced cryptographic schemes introduced in this paper are Fully Homomorphic Encryption and Functional Encryption. Fully homomorphic encryption supports manipulating the cipher texts and thus allows operations to be performed on them without losing its original meaning. Functional encryption on the other hand supports specific criteria based encryption/decryption process that helps in secured sharing information, for computing platforms such as the cloud.

ACKNOWLEDGEMENT

Nilotpal is thankful to the SPARK program of CSIR- Fourth Paradigm Institute, Bangalore for giving him the opportunity to carry out his major research project in the organization. The work is partially supported by the project ARiEES (CySeRO work package), funded by CSIR, India, under the 12th Five Year Plan.

REFERENCES

[1] Peter Mell, Timothy Grance; The NIST Definition of Cloud Computing; NIST Special Publication 800-145, 2011

[2] D S Bhilare, Nilotpal Chakraborty; Enhanced Security in Cloud Computing Environment, International Journal of Advanced Research in Computer Science and Software Engineering, Vol 3 Issue 9, September 2013.

[3] K Hamlen, Security Issues for Cloud Computing, Available online at: http://www.igi-global.com/chapter/security-issues-cloud-computing/

[4] R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169-177. Academic Press, 1978

[5] R. Rivest, A. Shamir and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM 21 (2): 120-126, 1978.

[6] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete alogarithms. In Advances in Cryptology—CRYPTO '84, volume 196 of Lecture Notes in Computer Science, pages 10-18. Springer-Verlag, 1985.

[7] Paillier, Pascal. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". *EUROCRYPT*. Springer. pp. 223–238, 1999, doi:10.1007/3-540-48910-X_16.

[8] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC '09*, pages 169-178, ACM, 2009.

[9] C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. http://crypto.stanford.edu/craig.

[10] Amit Sahai, Brent Waters; "Fuzzy Identity Based Encryption", Proceedings of Eurocrypt, 2005

[11] G K Patra, Nilotpal Chakraborty; Securing Cloud Computing Environment with the help of Fully Homomorphic Encryption, Journal of Computer Technology & Applications, ISSN: 2229-6964, Vol 4 Issue 3, December 2013.

[12] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Advances in Cryptology - EUROCRYPT'10, volume 6110 of Lecture Notes in Computer Science, pages 24-43. Springer, 2010. Full version available on-line from http://eprint.iacr.org/2009/616

[13] ZHANG-Tong, WU-Qi, LIU-Wen, CHENLiang, Homomorphism Encryption Algorithm for Elementary Operations over Real Number Domain, International Conference on Cyber-Enabled Distributed Computing and Knowledge Discover, DOI 10.1109/CyberC.2012.35.

[14] S.Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, Nickolai Zeldovich, Reusable Garbled Circuit and Succinct Functional Encryption, STOC'13, June 1–4, 2013, Palo Alto, California, USA.

[15] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. Fully homomorphic encryption without bootstrapping. In Innovations in Theoretical Computer Science (ITCS'12), 2012. Available at http://eprint.iacr.org/2011/277

[16] N. P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography - PKC'10*, volume 6056 of *Lecture Notes in Computer Science*, pages 420-443. Springer, 2010.

[17] J. S. Coron, D. Naccache, M. Tibouchi; Public Key Compression and Modulus Switching for Fully Homomorphic Encryption over the integers", Cryptology ePrint Archive, Report 2011/440, 2011. Available at http://eprint.iacr.org/