IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 6, Ver. III (Nov – Dec. 2015), PP 10-14
www.iosrjournals.org

A survey on Fully Homomorphic Encryption

Siddhi Khamitkar
Computer Engineering DepartmentMukesh Patel School of Technology Management and Engineering
NMIMS University
siddhi.khamitkar94@gmail.com

Abstract:*Cloud computing is an ever-growing field in today's era.With the accumulation of data and the advancement of technology,a large amount of data is generated everyday.Storage, availability and security of the data form major concerns in the field of cloud computing.This paper focuses on homomorphic encryption, which is largely used for security of data in the cloud.Homomorphic encryption is defined as the technique of encryption in which specific operations can be carried out on the encrypted data.The data is stored on a remote server.The task here is operating on the encrypted data.There are two types of homomorphic encryption, Fully homomorphic encryption and patially homomorphic encryption.Fully homomorphic encryption allow arbitrary computation on the ciphertext in a ring, while the partially homomorphic encryption is the one in which addition or multiplication operations can be carried out on the normal ciphertext.Homomorphic encryption plays a vital role in cloud computing as the encrypted data of companies is stored in a public cloud, thus taking advantage of the cloud provider's services.Various algorithms and methods of homomorphic encryption that have been proposed are discussed in this paper.*

Keywords:*Cloud computing, homomorphic encryption, security, fully homomorphic encryption, partially homomorphic encryption.*

I. Introduction

Homomorphic encryption refers to the operations carried out on the encrypted data.Cloud computing has security as its major issue.Homomorphic encryption is used to support simple aggregations, numeric calculations on the data.The basic concept of homomorphic encryption is to encrypt the data before giving it to the service provider.But this gives rise to a problem at the client's side.As the service provider needs to perform calculations on the data so as to respond to the client's request, key must be provided to the server to decrypt the data before executing the calculations which will surely affect the confidentiality of the data.Various concepts are being introduced in the paper.

One of the best mechanisms for security is that of fully homomorphic encryption which is also called as the cryptographer's holy grail.Focus is given on the construction of a generic runtime container capable of executing arbitrary encrypted programs.New features of homomorphic encryption are paid attention to.Fully homomorphic encryption is carried out on encrypted circuits as well for encrypted storage access with encrypted access and encrypted branching.

Fully Homomorphic encryption is the one in which "fully" means that there will be no restrictions on the manipulations that can be performed.Given the ciphertexts$c_1,c_2....c_t$that encrypt the plaintext $m_1,m_2....m_t$with a scheme under a key and given any efficiently computable function f that can calculate the ciphertext(or a set of ciphertexts)under that key.Thus, general computations on the encrypted data are permitted with no revealence of $f(m_1,m_2....m_t)$..

Section II(A) describes about homomorphic encryption.Section II(B) gives an overview of somewhat fully homomorphic encryption.Section II(C) explains Gentry's Fully Homomorphic Encryption.Section II(D) talks about recryption.Section II(E) explains bootstrapping.Section II(F) gives an overview of the BGV cryptosystem.Section III talks about potentials of homomorphic encryption in cloud whereas section IV gives the applications of Fully Homomorphic Encryption in cloud.On the basis of the study we draw conclusions and lastly, acknowledgements are mentioned in the paper.

II. Brief Description

A. Homomorphic Encryption

Homomorphism refers to the structure preserving traansformation between two sets where an operation on two members in the first set is preserved by the operation on the corresponding members in the second set.Let P and C be the sets.[7]

p1,p2εP.

t:transformation function t':reverse function. ⊕:operation.

The sysytem is said to be homomorphic if ∀p1,p2 ε P,(p1 ⊕p2)=t'(t(p1) ⊕t(p2)),(p1 ⊗p2)=t'(t(p1)⊗t(p2).Apply p1 and p2 in the range of C, thus applying some sort of encryption and having ⊕and ⊗performed by a third party. An algebraic cryptosystem can be described as a 6 tuple H1:{P,C,t,t',⊕,⊗} where:
P:plain text space C:cipher text space t and t':encryption and decryption functions ⊗and ⊕: operations to be carried out.

For noise reduction, a second tuple is introduced as H2:{P,C,t,t',⊗,⊕,r}.H2 has an additional reduction function r which is responsible for noise reduction. Consider the following example:
pεN: where P is one large prime number, consider 23. a,b:2 arbitrary integers,consider 4,6. a,bεN and are less than P. a:a+r*p. r:any random large integer. consider r1=100 and r2=200
Thus, encrypt and decrypt function are demonstatedbelow.Encrypted addition is shown by a'+b' while encrypted multiplication is shown by a'*b'[7].
a'=4+100*23=2304. b'=6+200*23=4606. a'+b'=2304+4606=6910 a'*b'=2304*4606=10607616
Now to decrypt, all we have to do is add a modulus p to the function.Thus giving us the values of a+b and a*b[7].
(a+b)=a+b+(r1+r2)*p mod p (a+b)=4+6+300*23 mod 23 thus a+b=10.
Similarly for multiplication the solution is:
(a*b)=a*b+a*(r2*p)+b*(r1*p)+(r1*r2)p mod p.
(a*b)=4*6+4*(200*23)+6*(100*23)+(100*200)*23 mod 23.
(a*b)=24

Thus from the values of a*b and a+b, the original values of a and b can be found out.Thus an example of homomorphic encryption is shown above[7].

B. Somewhat Fully homomorphic encryption

According to Smart Verkauterancryptosystem,somewhat fully homomorphic encryption is the one in which decryption works only as long as the cipher text noise is within certain limits of r[5].

Consider a scheme built on integers.p is the secret key whose size depends on the security criteria.It is an odd integer.To encrypt a bit m ε{0,1}, choose random integers q and r within a range such that 2r=p/2 and then it is set. c=Enc(m)=qp+2r+m.
We can retrieve the plaintext m by computing c mod p mod 2.In the above statement, c is the ciphertext, and q and r areintegers[4].
Let us consider two cipher texts c1 and c2, the computation will be as shown below: c1+c2=(q1+q2)p+2(r1+r2)+(m1+m2) To decrypt c1+c2,one needs to get m1+m2.This is possible only when the condition 2(r1+r2)+m1+m2≤p is verified.In this scenario, we have: c1+c2 mod p=2(r1+r2)+m1+m2.
m1+m2 can be verified by calculating c1+c2 mod p mod 2.Since we pick r such that 2r<p/2,the above condition is satsfied when the ciphertexts c1 and c2 are fresh ciphertext.Justincase if c1 is the sum of other two ciphertexts, the above condition will not be met and it is impossible to decrypt c1+c2.The scenario is even worse when multiplication comes into picture.This is when we have to manage the noise component which is the remaining random part.For e.g.:2(r1+r2).Keep the component under certain limit to ensure that decryption is possible.The noise component can be removed by Gentry's FHE i.e. Gentry's Fully Homomorphic Encryption[4].

C. Gentry's FHE

In his proposal, Gentry uses ideal lattices to explain his method.He considers a ciphertextψ in the form v+x where v is the ideal lattice and x is the error or offset vector encoding the plaintext π. The operations are interpreted to be carried out in a ring Z[X]/f(x) where addition and multiplication is done using ring operations:ψ←ψ1+ψ2 or ψ←ψ1xψ2 and induce multiplication or addition of the underlying plaintext.This scheme introduced by Gentry itself improves on the prior work that is done in this field.ε1 is the security parameter that is set based on the decisional version of the closest vector problem for ideal lattices for ideals in a fixed ring[2].

ε1is only for shallow circuits as with the complexity, the error grows with addition and highly with multiplication.We can refresh the ciphertext of large length to a ciphertext of a shorter length to decrypt it via homomorphism, and this exactly, is the idea behind bootstrapping[2].

Suppose, we have ε that is bootstrappable, with the plaintext space P being{0,1}, and the circuits being boolean.Consider we have a ciphertextψ1 encrypting π under pk1, which we want to refresh.Let sk1 be the secret key for pk1 so as to decrypt π homomorhically.sk1 is encrypted under a second public key pk2.Let sk_1j be the secret key bits that are encrypted.Now consider the algorithm for recrypt.

D. Recrypt

Set $\psi_1 j \leftarrow \text{Encrypt}_{\varepsilon}(pk_2, D_{\varepsilon}, ((\bar{sk}_1 j), \bar{\psi}_1 j))$.

We now explain the above functions: Evaluate takes in the bits on sk1 and ψ1, each encrypted under pk2.ε is then used to evaluate the circuit homomorphically.We can thus conclude that the output ψ2 is an encryption under pk2 of $Decrypt_{\varepsilon}$(sk1,ψ1) = π^2.D_{ε} removes the error vector associated with the ciphertext.However, evaluate introduces another error that has a shorter length[2].

Note:If there are two copies of D_{ε}connected via a NAND gate, ε is said to bootstrappable[2].

E. Bootstrapping

A circuit is said to be bootstrappable when it not only evaluates its encryption circuit but also augmented versions of it to carry non trivial operations[2].Let ε.Letε be a tuple such that ε :{Keygen, Encrypt, Decrypt,Evaluate} be a homomorphic encryption scheme.For every security parameter(λ),let C_{ε}be a set of circuits with consideration to which ε is correct.εis said to be bootstrappable if $D_{\varepsilon}(\lambda) \subseteq C_{\varepsilon}(\lambda)$ holds true for every λ where $C_{\varepsilon}$and$D_{\varepsilon}$are functions[3].

Suppose that ε is bootstrappable and can handle four functions:A decryption function $D_{\varepsilon}$of polynomial size λ as well as D_{ε}augmented by the add, subtract or multiply gate modulo 2.If these four circuits are in F_{ε}, then ε^+ can be constructed from ε.Thisε^+ is fully homomorphic.

We then perform recrypt function on the data as described in section D.

The aim here is to perform new operations on the underlying messages and not just obtain the new encryption.Ifε can handle D_{ε}augmented by some gate consider Add.Cal this augmented circuit $D_a$dd.If c1 and c2 encrypt m1 and m2 respectively under pk1, we compute c¯1 and c¯2 as described earlier as ciphertext encrypting the bits of ciphertext pk2, then what we have under pk2 is an encryption ofm1⊕m2. c← $Evaluate_{\varepsilon}$(pk_2,$D_a$dd,sk_1,c¯1,c2)

By following a recursive pattern for the same, we get fully homomorphic encryption.The public key in ε^+ consists of a series of public keys {pk_1,....pk+ l + 1} and a chain of encrypted secret keys {c_1,...sk_l} where sk_i is encrypted under pk_i+ 1.To evaluate f in ε^+, f is expressed as a circuit, topologically arranging its states into levets and stepping through the levels sequentially.For a gate at the i+ 1_th level.Let this be applied to an add gate.

input:secret key($\bar{sk}_i$) output: wires at level iunderpk_i

We homomorphically evaluate $D_a$ddto get ciphertext at $pk_($$i_+$1) associated to a wire at level i+1.The output is finally sent as an output to the output wire of f[6].

F. BGV cryptosystem

BGV is an assymetric encryption technique included as a method in somewhat fully homomorphic encryption.It is based on ideal lattices and has two versions:

1) **Learning With Errors(LWE)**
2) **Ring-Learning With Errors(R-LWE)**

Learning With Errors deals with integer vectors while the Ring Learning With Errors deals with integer polynomials.The security of R-LWE is linked to the hardness of the decisional R-LWE problem.The R-LWE consists of the difference in the distribution of (a_i,b_i) sampled uniformly across $Z_q^n X Z_q$ in the ring $A = Z_q^n / f(X)$ and a distribution of(a_i,<a_i,s>+e_i) where:

$a_i$and s:sampled from Z_q^n e^i:sampled according to Gaussian distribution.

Focus is given on the polynomial version of BGV as it is more accurate.

We consider the polynomial ring A=Z[X]/F(X) where:

F(X):cyclomatic polynomial of degree d=2^k and a chain of odd modulii q1<.....$q_L$and their corresponding subrings A_qi= A/q_iAof polynomials of A with integers coefficients in the range[-q_i/2,q_i/2]

In practice, elements in $A_q$iwill be the polynomials represented by d-vectors of their co-efficients[1].

III. Potentials Of Homomorphic Encryption In Cloud

If all the data is stored in the encrypted form in cloud, the user will fail to carry out computations on data without first decrypting it, though it will provide a higher level of security.Thus it is the responsibility of the cloud provider to decrypt the data,performing the computation first and then sending the result to the user.Homomorphic encryption allows the user to carry out any arbitrary operation on the hosted data without the cloud provider learning about the users data.This means that computation is done on the user's data without prior decryption.It allows the transformation of ciphertext C(m) of a message m to C(f(m)) which is a computation/function of message m.

The idea was first suggested by Rivest,Adleman and Dertouzos in 1978 referred to as privacy homomorphism.RSA which was formulated by Rivest, Shamir and Adleman had multiplicative homomorphism.In the coming 30 years, Goldwasser and Micali ,Elgamal and Paillier came up with partially homomorphic cryptosystems[8].

The first fully homomorphic encryption model was suggested by Craig Gentry and is discussed in the previous section i.e. section II.C.

IV. Applications Of Fully Homomorphic Encryption In Cloud

Fully Homomorphic Encryption is an important factor for implementing security in cloud.Generally stating, it is possible to keep the secret key that can decrypt the result of the calculation and still outsourcing the calculation to the cloud server.A database server can communicate with the client as shown in the figure below[10].

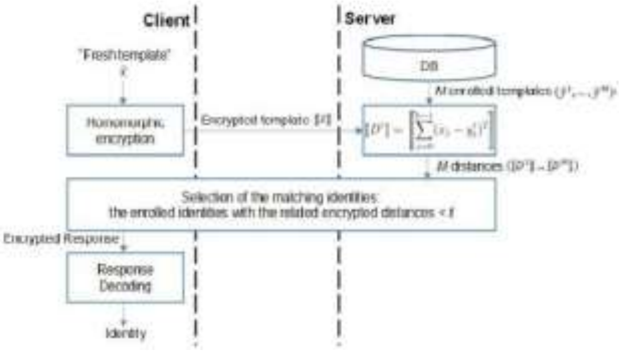

Fig 1: A database server and client implementing homomorphic encryption[10].

Similarly, the cloud computing scenario will be the one as shown in the figure below:

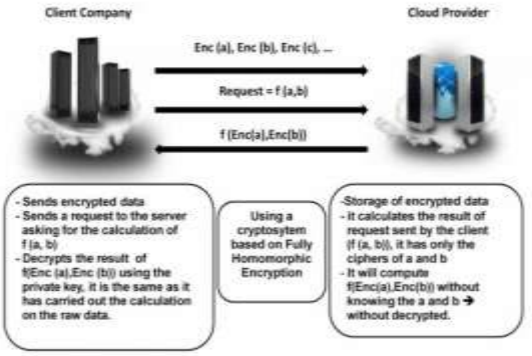

Fig 2: Implementation of FHE in a cloud server[10].

V. Conclusion

Thus a detailed survey on Fully Homomorphic Encryption has been carried out in this paper.Homomorphic encryption is being highly applied in the field of cloud computing for security purposes.Craig Gentry was the first researcher to come up with the idea of Fully Homomorphic Encryption in the year 2009 and suggested that homomorphic encryption is bootstrappable.A detailed description of Gentry‘s algorithm, the BGV technique and several other important concepts have been studied and explained in this paper.The applications of Fully Homomorphic Encryption in the field of cloud have also been studied.We can thus conclude that Fully Homomorphic Encryption has played a vital role in the security and the confidentiality of the data and is being applied on a large scale in various applications over the cloud.

VI. Acknowledgements

I would like to thank my professors and mentor for their immense support and encouragement.I would also like to thank my peers for their suggestions for improvement and constructive criticism.

References

[1] Simon Fau,CarolaineFontaine,CarlosAnguilar-Melchor,GuyGogniat, "Towards practical program execution over fully homomorphic encyption schemes". IEEE,Eighth international conference on P2P, parallel, grid, cloud and internet computing,2013.

[2] Craig Gentry, "Fully homomorphic encryption using ideal lattices",Proceedings of the 41st annual ACM symposium on theory of computing,2009.

[3] C.Gentry,V.Vaikuntanathan "Fully Homomorphic Encryption over the integers ",in EUROCRYPT(LNCS)vol.6110,2010.

[4] C.AnguilarMelchor,S.Fau,C.Fontaine,G.Gogniat and R.Sirdey "Recent advances in homomorphic encryption:A possible future for signal processing in the encrypted domain,IEEE signalprocess.Mag.,30(2).,108-117 ",2013

[5] Henning Perl,Michael Brenner and Matthew Smith,"Poster:An implementation of the fully homomorphic Smart Vercauterancryptosystem",ACM conference on Computer and Communications Security,page 837-840,2011.

[6] Craig Gentry, "Computing Arbitrary functions of the encrypted data",Communications of the ACM,vol 53 n0.3,page 97-105,2010.

[7] Michael Brenner,JanWiebelitz,Gabriele von Voigt and Matthew Smith,"Secret Program Execution in Cloud Applying Homomorphic Encryption ",5th IEEE International Conference on Digital Ecosystems and Technologoes,2011.

[8] AderemiA.Atayero and OluwaseyiFeyisetan,"Security issues in cloud computing:The potentials of Homomorphic Encryption",Journal of Emerging Trends in Computing and Information Sciences,vol.2 no.10,2011.

[9] Iram Ahmad and Archana Khandekar,"Homomorphic Encryption method applied to Cloud Computing",International Journal of Information and Computation Technology,vol 4,no.15,2014 .
[10] Shashank Bajpai and PadmijaSrivastava,"A Fully Homomorphic Encryption Implementation on Cloud Computing",International Journal of Information and Computation Technology,vol 4,no.8,2014 .
[11] Chen Yu Tsai and DJ Guan,"Applications of homomorphic encryption",Masterthesis,National Sun Yat-Sen university,2013.
[12] Q.Liu,C.Tan,J.Wu and G.Wang,"Efficient Information Retrieval for Ranked Queries in Cost Effective Cloud environments",proc.INFOCOM,2012.