北京理工大學

本科生毕业设计（论文）外文翻译

外文原文题目：Fully homomorphic encryption:searching Over encryption cloud data

中文翻译题目：全同态加密：云加密数据搜索

基于国产密码算法的云计算网络信息传输认证系统设计与实现

The Subject of Undergraduate Graduation Project (Thesis) of Beijing Institute of Technology

学　　院：计算机学院

专　　业：软件工程

学生姓名：侯添久

学　　号：1120161912

指导教师：闫怀志

Fully homomorphic encryption: Searching over encrypted cloud data

Ahmed EL-YAHYAOUI

Information Security Research Team, CEDOC ST2I ENSIAS, Mohammed V University in Rabat, Rabat, Morocco

ahmed_elyahyaoui@um5.ac.ma

Mohamed Dafir EC-CHRIF EL KETTANI

Information Security Research Team, CEDOC ST2I ENSIAS, Mohammed V University in Rabat, Rabat, Morocco

dafir.elkettani@um5.ac.ma

Abstract

Searching over encrypted data is a very important property that can offer some encryption schemes. It means performing queries on encrypted data in the absence of purpose to decryption, which permits to preserve confidentiality of sensitive data. Private Information Retrieval (PIR) is an essential protocol when selected information from remote databases is wanted to be done in secrecy. Fully homomorphic encryption is a revolutionary domain of cryptography that allows processing encrypted data without the need of any prior decryption, thus generating an encrypted result that corresponds the result of operations performed on the plaintext. In this paper we will exploit the important property that can offer the powerful fully homomorphic encryption to show how to realize a PIR protocol and how to search over encrypted data in a cloud context.

Keywords: search, encrypted data, PIR, protocol, cloud, database, homomorphic encryption.

1. Introduction:

In a context of cloud computing and outsourcing data storage. Many clients' uses cloud services to reduce operational costs of backups and infrastructure maintenance. A client stores confidential information, on outlying and anonymous cloud environment, in an encrypted form to avoid untrusted servers and third part to leak confidentiality of his sensitive data.

Law imposes sometimes encryption of some specific outsourced data. For example, legislation in many countries around the word states that electronic health records (EHRs) must be encrypted. Normal utilization of data becomes a challenge after encryption. Searching over encrypted data is an important property that can offer some encryption schemes. It is highly needed if the client want to query his encrypted database on the cloud server.

Searching over encrypted data (figure 1) was first introduced by Song, Wagner and Perrig [1]. The scheme from [1] is a searchable symmetric encryption (SSE) which is symmetric key cryptography based. It allows a single client to write and read data, i.e. only the secret key holder is able to create searchable ciphertexts and trapdoors. A second category of searchable encryption schemes will be risen after a pioneering work of Boneh et al in 2004 [2], who invented a public key encryption with keyword search (PEKS) scheme. It is a public key encryption based, for which the private key can decrypts all messages encrypted under the corresponding public key, i.e. It permits multi-clients to write (to encrypt) and only a single user to read (to decrypt).

Private Information Retrieval (PIR) [3] (Figure 2) is a protocol that allows a user to obtain an element of a database by hiding from it what element it is. While this problem admits a trivial solution – The client retrieves the entire database and query with a perfect privacy – this technique is not applicable for large databases because of the communication complexity.

Combining searching over encrypted data with private information retrieval enhances the cloud security and offers more protection to sensitive data. This solution gives more flexibility to clients to deal with their outsourced cloud data as it allows the cloud computing to do more operations on encrypted database.

Fully homomorphic encryption is a powerful tool that can allow searching over encrypted data and realizing PIR protocols. It enables cloud server to search blindly on client's encrypted databases without decrypting it or acquiring any knowledge about the plaintext data or the searched query. The first apparition of fully homomorphic encryption was in 1978 with Rivest,

BDCA'17, March 29-30, 2017, Tetouan, Morocco

ACM ISBN 978-1-4503-4852-2/17/03…$15.00
http://dx.doi.org/10.1145/3090354.3090364

Adleman and Dertozous under the name of privacy homomorphism [4], but privacy homomorphism stayed a conjecture without an effective solution until 2009. In his thesis, Craig Gentry presented the first semantic secure privacy homomorphism under the name of fully homomorphic encryption scheme [5]. The scheme from [5] uses a bootstrapping theorem to reduce the noise generated after processing ciphertexts. The cleartext space of Gentry's scheme [5] is the binary field, such a space allows us to evaluate any circuit on encrypted data using homomorphic capacities of the scheme. Several works followed Gentry's breakthrough and proposed new fully homomorphic encryption schemes [6], [7], [8]… some of this new constructions tried to use large cleartext spaces [9], [10] to reduce encryption cost.

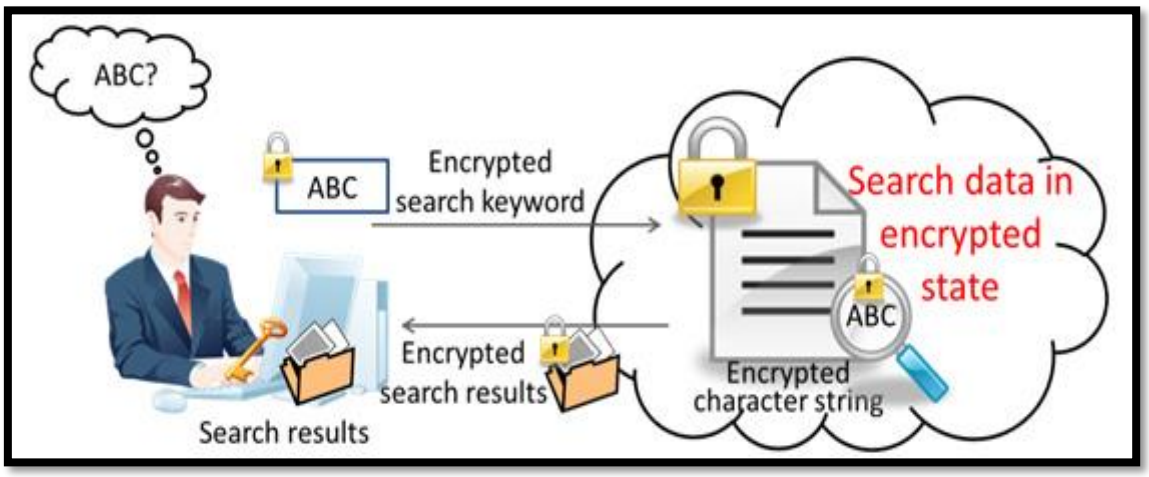

Figure 1: Search over encrypted data

In this paper, we propose an efficient search solution and a PIR protocol based on fully homomorphic encryption. For that reason we will focus on binary circuit based fully homomorphic encryption schemes. I.e. cryptosystems for which the cleartext space is {0,1} with XOR and AND operations.

2. Notations

In the following, the notation $\mathcal{E}(m)$ will denote the encryption of the message m and the notation $\mathcal{D}(c)$ will denote the decryption of the ciphertext c. Intuitively, if we have $c = \mathcal{E}(m)$ then $m = \mathcal{D}(c)$.

3. Homomorphic encryption

In its large sense, a homomorphic encryption is a cryptosystem that allows a user, in addition to encryption and decryption operations, to perform computations on encrypted data. If the encryption scheme gives permission to a limited number of computations, it said partially homomorphic. For example, the unpadded RSA [11] cryptosystem is supposed to be multiplicatively homomorphic, i.e. for two cleartext m_1 and m_2 we have $\mathcal{E}(m_1) = {m_1}^e \ mod\ N$ and $\mathcal{E}(m_2) = {m_2}^e \ mod\ N$ such that N is the RSA modulus and e is the public key exponent, so $\mathcal{E}(m_1).\mathcal{E}(m_2) = {m_1}^e.{m_2}^e mod\ N = (m_1.m_2)^e mod\ N = \mathcal{E}(m_1.m_2)$. RSA allows us to perform only multiplications on ciphertext but no addition, an example of an additively homomorphic encryption scheme is Paillier's cryptosystem [12], if the public key is the modulus N and the base g, then the encryption of a message m is $\mathcal{E}(m) = g^m r^N mod\ N^2$, for some random $r \in \{1, \dots N-1\}$. The homomorphic property of Paillier's scheme is then $\mathcal{E}(m_1).\mathcal{E}(m_2) = (g^{m_1}{r_1}^N).(g^{m_2}{r_2}^N) mod\ N^2 = g^{m_1+m_2}(r_1 r_2)^N\ mod\ N^2 = \mathcal{E}(m_1 + m_2)$.

In the other hand a fully homomorphic encryption scheme is a cryptosystem that allows aclient, in addition to encryption and decryption operations, to perform any calculation on his encrypted data without decrypting it. Suppose that we have two cleartext m_1 and m_2, let $c_1 = \mathcal{E}(m_1)$ and $c_2 = \mathcal{E}(m_2)$ be its ciphertexts respectively. A fully homomorphic encryption permits us to obtain $c_1 + c_2 = \mathcal{E}(m_1 + m_2)$ and $c_1 \times c_2 = \mathcal{E}(m_1 \times m_2)$. For that reason, a user can delegate his complex computations to a remote cloud, which holds unlimited computation powers, to do it in his place.

Mathematically, a fully homomorphic encryption scheme is a quadruplet of polynomial algorithms $(Gen, Enc, Dec, Eval)$ verifying:

- $Gen(\lambda)$: Is an algorithm of key generation , takes as input a security parameter λ and outputs a public and secret keys (pk, sk).
- $Enc(m, pk)$: Is an encryption algorithm, takes as input a plaintext m and a public key pk and outputs a ciphertext c.
- $Dec(c, sk)$: Is a decryption algorithm, takes as input a ciphertext c and a secret key sk and outputs a plaintext m.
- $Eval(C, c_1, \dots, c_n)$: Is an evaluation algorithm, takes as input a circuit C and ciphertexts $c_1, \dots, c_n$ and verifies $Dec(Eval(C, c_1, \dots, c_n), sk) = C(m_1, \dots, m_n)$. Anyone can evaluate $Eval$, since it does not require the secret key sk.

In the following, our scheme will be a circuit-based cryptosystem that allows a client to encrypt a cleartext bit by bit.

In addition, we will assume that we have a database DB, with n encrypted entries $c_1, c_2 \dots c_n$, stored in a remote cloud-computing server.

4. Test of equality of two encrypted messages

With a fully homomorphic encryption scheme, one can test if two encrypted messages are equals or not without decrypting it. In this section, we will develop the operator $equal(c_1, c_2)$, which takes in input two ciphertexts $c_1 = \mathcal{E}(m_1)$ and $c_2 = \mathcal{E}(m_2)$ and permits to verify if m_1 and m_2 are equals without decrypting the given ciphertexts. To establish this operator we will need the following two operations to be done on ciphertexts:

4.1. Inversion

Inversion of an encrypted bit is an operation that permits to transform the encryption of a single bit to the encryption of its inverse based only on ciphertext. This operation can be realized from a fully homomorphic encryption scheme as:

Assume that $\sigma \in \{0,1\}$ and that $c = \mathcal{E}(\sigma)$, we will note $\bar{c} = \mathcal{E}(\bar{\sigma})$ such that $\bar{\sigma} = \sigma \oplus 1$ is the inverse of σ.

$\bar{c}$ can be obtained from c by calculating $\bar{c} = c + \mathcal{E}(1) = \mathcal{E}(\sigma) + \mathcal{E}(1) = \mathcal{E}(\sigma \oplus 1) = \mathcal{E}(\bar{\sigma})$.

4.2. Ones' complement

The ones' complement of a binary number is defined as the value obtained by inverting all the bits in the binary representation of the number. It is a kind of inversion of multiple bits at the same time. To compute the ones' complement of a bit-by-bit encrypted input, we carry out the inversion of each encrypted bit.

Let m be a binary cleartext (which is not necessarily a bit) encrypted to $C = \mathcal{E}(m)$ and $\bar{C} = \mathcal{E}(\bar{m})$ such that $\bar{m}$ is the ones' complement of m. We have $\bar{C} = \mathcal{E}(m) + \mathcal{E}(111 \dots 1) = \mathcal{E}(m \oplus 111 \dots 1) = \mathcal{E}(\bar{m})$.

4.3. Equality

As it precedes $equal(c_1, c_2)$ takes two ciphertexts as input. $equal$ will return an encrypted result which once decrypted we obtain a single bit. If the bit is 1 that means m_1 and m_2 are equals, else m_1 and $m_2$are not equals.

To test equality of m_1 and m_2, only by using c_1 and c_2 we will proceed as follows:

1. Compute the bitwise addition $c_1 \oplus c_2 = \mathcal{E}(m_1 \oplus m_2)$

If $m_1 = m_2$ we will obtain $C = c_1 \oplus c_2 = \mathcal{E}(000 \dots 0)$ otherwise, we will have among the encrypted bits of $c_1 \oplus c_2$ at least one equal to 1.

2. Compute the ones' complement of $c_1 \oplus c_2$, which is $\overline{c_1 \oplus c_2}$.
3. Multiply all the encrypted bits of the obtained result. As a consequence, we will obtain only an encrypted bit (0 or 1).
4. If the obtained result $equal(c_1, c_2) = \mathcal{E}(1)$ then we have $m_1 = m_2$, else $equal(c_1, c_2) = \mathcal{E}(0)$ then $m_1 \neq m_2$.

5. Search over encrypted data

Security is still a major inhibitor of cloud computing, when companies are testing cloud applications for storage and databases. Storing data in a cloud database in an encrypted form is very desirable today. The encryption allows us to protect the sensitive contents in the data, but this usually implies that one has to sacrifice functionality for security. Searching over encrypted data is the solution that should be of interest. It allows a cloud server to search blindly on client's encrypted data, based on a 'trapdoor' - a token that contains keywords to be searched for - sent by the client. Of course, client must also encrypt the keywords along the encrypted data before uploading them, which will surely increase the overhead. For 'blindly' it means the cloud server does not acquire any unnecessary knowledge about the searched keywords and the encrypted data, during the entire query process.

Fully homomorphic encryption allows us to query an encrypted database without the need for decryption key. In addition to doing computations on ciphertexts, it permits to search blindly over encrypted data.

Suppose that we have a database with n entries $c_1, c_2 \dots c_n$, to look for the existence of an encrypted message $C = \mathcal{E}(m)$ among these encrypted entries we proceed as follows: After sending the query to the server, it proceeds as follows:

-It extracts the encrypted message C from the query and tests its equality with each entry of the encrypted database.
Because of this phase, we will obtain n encrypted bits $b_1 = equal(C, c_1)$, $b_2 = equal(C, c_2)$,… $b_n = equal(C, c_n)$.
-If C does not exist in the database, the n encrypted bits will be all equals to an encrypted 0 i.e. $b_1 b_2 b_3 \dots b_n = \mathcal{E}(000 \dots 0)$. Else we will obtain at least one $b_i = \mathcal{E}(1)$ for a given $i \in [\![1, n]\!]$, i.e. $b_1 b_2 b_3 \dots b_i \dots b_n = \mathcal{E}(000 \dots 1 \dots 0)$.

So to verify the existence of C in our database, the server should test the equality of the result $(b_1 b_2 b_3 \dots b_n)$ with $\mathcal{E}(000 \dots 0)$. i.e. it computes $SEARCH = equal(b_1 b_2 b_3 \dots b_n, \mathcal{E}(000 \dots 0))$.

- The obtained result will be inverted and sent to the requester. i.e. $\overline{SEARCH}$
After receiving the $\overline{SEARCH}$ encrypted bit, the requester will decrypt it. Two possible results are waited:
If he finds 0, so the searched query does not existed. Otherwise, hi will finds 1 which means that the searched query exists.

6. Private Information Retrieval

Assume that a client has queried a database and found a positive result ensuring that the searched data exists in the database. The protocol PIR (figure 2) allows him to retrieve his wanted information's in a safer way and without the cloud server being able to determine which element was selected. PIR protocols can be combined with searchable encryption schemes to obtain a more secure protocol which permits a clients to search on encrypted cloud data and retrieve it privately. This solution functions very well when outsourced data are sensitive. Fully homomorphic encryption is able to allow both: searching over encrypted data and private retrieval of wanted database entries.

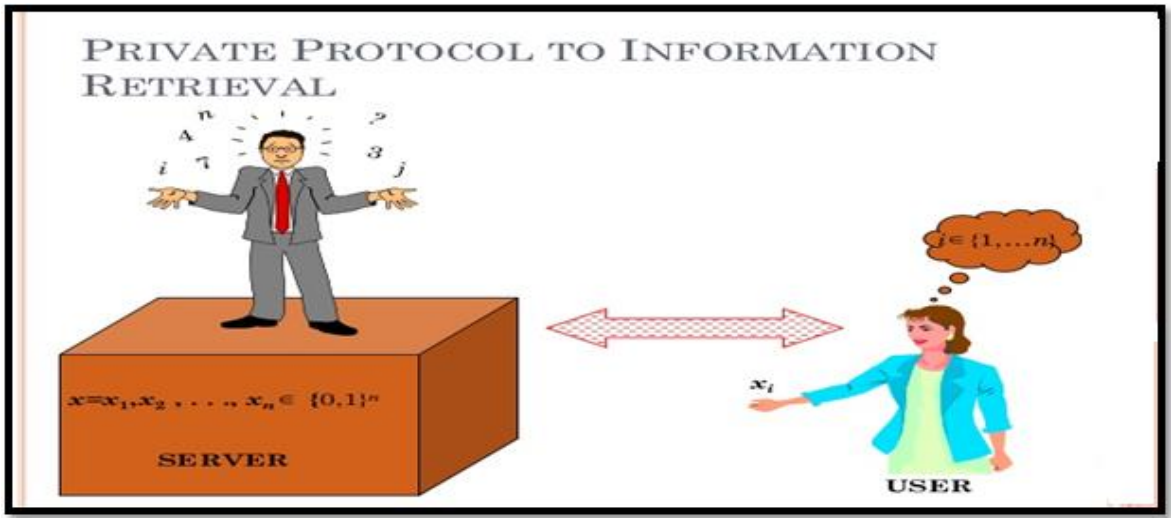

Figure 2: Private Information Retrieval Protocol

6.1. PIR from homomorphic encryption

PIR is realizable from homomorphic encryption. Assume that the database, with n entries $e_1, e_2, \dots, e_n$, is organized such that each entry is indexed by a unique integer. The entries in the database are not supposed to be encrypted, they are simply integers. We want to get the $k\ th$ element without the knowledge of the database owner. Suppose that we have a fully homomorphic encryption scheme.

The client starts by computing ciphertexts c_i, where c_i is an encryption of 0 except for i=k, in this case c_k is an encryption of 1. Because of our cryptosystem is a probabilistic scheme, 0 can be encrypted in several different ways. Therefore, all c_i are different. The client sends to the cloud server the query $(c_1, c_2, \dots, c_n)$. The database administrator calculates $c_1e_1 + c_2e_2 + \cdots + c_ne_n$ and returns this result to the client. He then calculates $\mathcal{D}(c_1e_1 + c_2e_2 + \cdots + c_ne_n)$ and retrieves e_k. In fact, we have $c_1e_1 + c_2e_2 + \cdots + c_ne_n = \mathcal{E}(0)e_1 + \mathcal{E}(0)e_2 + \cdots + \mathcal{E}(1)e_k + \cdots + \mathcal{E}(0)e_n = \mathcal{E}(0 * e_1 + 0 * e_2 + \cdots + 1 * e_k + \cdots + 0 * e_n) = \mathcal{E}(e_k)$. (See figure 3).

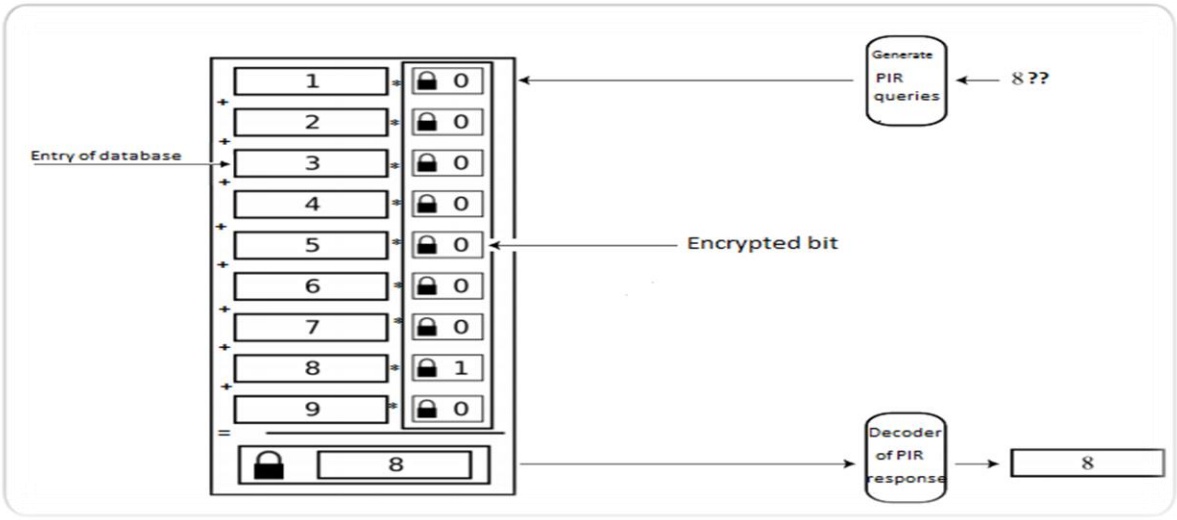

Figure 3: PIR from homomorphic encryption

7. Conclusion

In this paper, we treated the problem of searching over encrypted data and private information retrieval (PIR) protocol; all operations are done on a stored database in a remote cloud server. In the situation of an encrypted database, combining searching over encrypted data and PIR in one protocol is highly appreciated for a cloud context. Our approach to collaborate these two techniques is fully homomorphic encryption based. All operations are done, easily, under a cryptosystem that can allow computing on encrypted data without prior decryption.

Bibliography

[1] D. Song , D. Wagner et A. Perrig, «Practical Techniques for Searches on Encrypted Data,» *IEEE Symposium on Security and Privacy*, pp. 44-55, 2000.

[2] D. Boneh, G. Crescenzo, R. Ostrovsky et G. Persiano, «Public key encryption with keyword search,» chez *EUROCRYPT (LNCS)*, 2004.

[3] B. Chor, O. Goldreich, . E. Kushilevitz et M. Suda, «Private information retrieval,» *In Proc. of the 36th Annu. IEEE Symp. on Foundations of Computer Science Pages 41–51, 1995. Journal version: J. of the ACM, 45:965–981, 1998.*

[4] R. Rivest, L. Adleman et M. Dertouzos, ««On Data Banks and Privacy Homorphisms,»,» *In Foundataions of Secure Computataion, Academic Press*, pp. 169-179, 1978.

[5] C. Gentry, «A fully homomorphic encryption scheme,» https://crypto.stanford.edu/craig/craig-thesis.pdf, September 2009.

[6] M. van Dijk, C. Gentry, S. Halevi et V. Vaikuntanan, «« Fully homomorphic encryption over the integers,»,» *Cryptology ePrint Archive, Report 2009/616, 2009. http://eprint.iacr.org/..*

[7] Z. Brakerski, C. Gentry et V. Vaikantanathan, ««« Fully Homomorphique Encryption without Bootstrapping,»,» *Available at http://eprint.iacr.org/2011/277..*

[8] J. Fan et F. Vercauteren, ««Somewhat Practical Fully Homomorphic Encryption,»,» *Available at http://eprint.iacr.org/2012/144. .*

[9] J. Kim, M. Sung Lee, A. Yun et J. Hee Cheon, «“ CRT-based Fully Homomorphic Encryption over the Integers”,» *Available at http://eprint.iacr.org/2013/057.*

[10] A. Kipnis and E. Hibshoosh, « « Efficient Methods for Practical Fully Homomorphic Symmetric-key Encrypton, Randomization and Verification »,,» *Cryptology ePrint Archive, Report 2012/637..*

[11] R. Rivest, A. Shamir et L. Adleman, «A Method for Obtaning Digital Signatures and Public Key Cryptosystems,» *Commununications of ACM, Vol. 21, pp. 120-126, April 1978..*

[12] P. Paillier, «Public-key cryptosystems based on composite degree residuosity classes.,» chez *18th Annual Eurocrypt Conference (EUROCRYPT'99)*, Prague, Czech Republic, 1999.

[13] D. Boneh, E. Goh et K. Nissim, «Evaluating 2-DNF formulas on ciphertexts. In Kilian, J., ed.: Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings. Volume 3378 of Lecture,» chez *Theory of Cryptography, Second Theory of Cryptography Conference*, Combridge, February 10-12, 2005.

全同态加密：云加密数据搜索

摘　要

对于加密数据的搜索是一个非常重要的特性，因为它可以提供一些加密场景的解决方案。 同时，它也意味着可以在对加密数据进行查询时不需要解密出相应的明文数据，这样可以在很大程度上保证使得敏感数据的不会被泄露。个人信息检索协议(PIR)是一个必不可少的约定，当我们需要的信息是从远程数据库中查询的得到时，在这整个处理过程中是希望可以保证数据的安全性，全同态加密技术是密码学上具有革命性的研究领域，因为它允许使用者直接操作加密的数据而不需要对密文数据进行解密，然后对于使用密文数据操得到的结果与直接使用明文进行相应操作得到的结果是等价的。在本论文中，我们将会使用到一个非常重要的特性它可以提供全同态加密来展示如何实现一个 PIR 协议以及如何在云环境中对密文数据进行搜索。

关键词：搜索；加密数据；PIR；协议；云计算；数据库；全同态加密技术

目　录

第一章 介绍

在云计算环境中，我们会将数据放到远程的云服务器上进行存储，非常多的用户都会选择使用云服务以此来减少运行时数据备份以及基础设施维护的开销，某个客户端可以将敏感的信息进行加密，然后将数据以密文的形式存储在某些公开的云环境中，以此来避免那些不受其信任的服务和第三方得到其数据，从而导致他自己的隐私数据发生泄漏。

在法律上，有时也会强制性要求对于某些某些特殊的数据必须进行加密，比如：许多国家围绕法律这个词强调 EHR 必须被加密，正常地使用这些加密后的数据变得极具挑战。对于加密数据的搜索是一种重要的特性，它可以提供一些数据加密的解决方案。这种目前也是用户具有强烈的需求，尤其是当用户想查询他那些存放在远程云服务器上的隐私数据时。

对于加密数据的搜索首次是由 Son,Wagner 和 Perrig 三人提出的，当时提出的方案是对称加密(SSE)的可搜索性，此方案是以对称密码技术为基础的。它可以允许某个客户端读写数据，并且只有那些密钥持有者可以创建可用于搜索的密文。第二类对于密文搜索的解决方案在 Boneh 等人的做了开创性工作之后被提出，他提出了一种基于关键词搜索(PEKS)的加密搜索方案，此方案是基于非对称密码技术的加密方案，所有消息都会使用公共的密钥钥进行加密，只有相应的私钥才可以对其进行解密，这种方案允许多个客户端对数据进行写操作（加密），但是只会有一个客户端对数据进行读取操作（解密）。

私人信息检索需要允许用户获取数据库的某个属性，该属性又需要通过数据库的某些隐藏属性获取。虽然这种解决方案存在着一些瑕疵，但是对于客户端检索整个数据库并且进行相应隐私数据查询是非常高效的，不过这种方案不适用于大型数据库，因为大型数据库的通信过于复杂，此方式将会增大双方通信的复杂性。

将对加密数据的搜索与私人数据检索相结合能够提升云计算的安全性，并且也可以为敏感数据提供更多强有力的保护。同时这个解决方案也可以为客户提供更大的灵活性来应对他们使用云来存储和管理数据上遇到问题，因为它允许云服务器在加密的数据上执行更多其他的操作。

全同态加密是一种非常强大的技术，因为它可以允许对加密的数据进行搜索并实现 PIR 协议，它可以使云服务器任意的搜索任何用户加密的数据而不需要解密数据获得相应的明文，即不需要得到任何有关明文数据的信息或相应的搜索查询的信息。同态加密技术想法第一次提出是在 1978 年，由 Rivest，Adleman 和 Dertozous 三人当时以隐私同态的名字被提出的，但是当时隐私同态仍然只是一个猜想而没有一个实际有效可行的方案。直到 2009 年，在 Craig Gentry 的文章中，他提出了第一个语义安全的完全同态下隐私同态加密的技术方案，这种方案主要是通过使用 bootstrapping 理论来减小在对加密数据进行处理时产生的噪音，Gentry 的明文空间是用一种二进制的字段表示，在此方案下的同态加密技术，就类似于在一个空间中，它允许我们可以在加密的数据上评估任何的环。一些紧随着 Gentry 之后的研究取得了突破并且提出了新的完全同态加密的解决方案，这些新的方案都是在通过尽可能的使用更大的明文空间来减小加密时的消耗的目标下实现的。

下图展示了一个基于关键字的加密数据搜索流程，首先用户输入自己想进行搜索的关键字，然后对输入的关键字进行加密操作，再将密文通过网络发送到远程的云服务器上，云服务器接收到相应的密文后，根据密文进行搜索，得到搜索的密文结果，最后将搜索得到的密文结果返回给用户，用户解密拿到自己想要的结果。

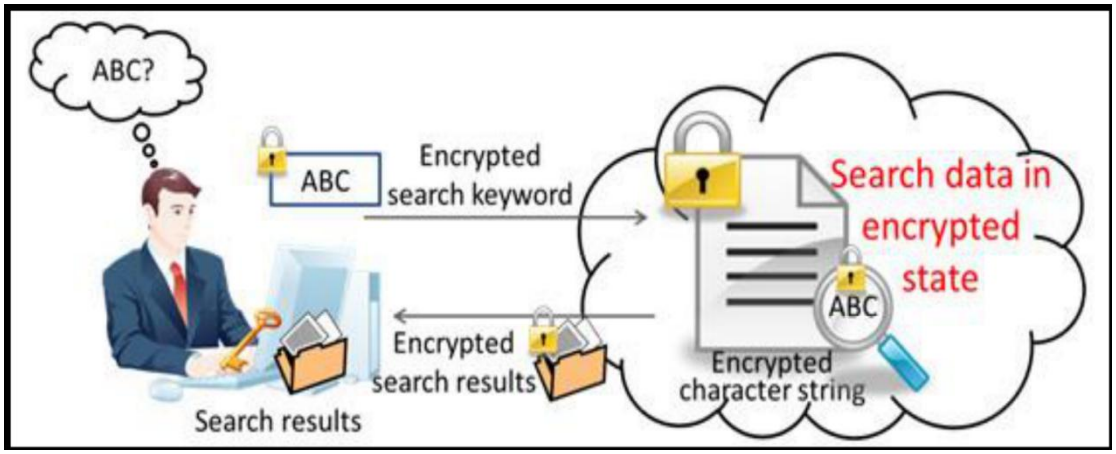

图 1-1 云上加密数据的搜索

在本章节中，我们提出了一种有效的对于密文数据的搜索解决方案和基于全同态加密的 PIR 协议。因次，我们主要将关注于基于全同态加密中的二进制环，即密码系统使用异或和与操作使得整个明文空间处于{0,1}。

第二章 符号标记

在本论文中，我们将会使用一些符号来表示相应的含义，使用符号 $\varepsilon(m)$ 表示对明文消息 m 使用 ε 函数进行加密操作并得到密文 c。使用符号 $\mathrm{D}(c)$ 表示对密文 c 使用 D 函数进行解密操作并得到明文 m。通常情况下，如果我们密文 c 与明文 m 有两个关系，它们可以表示为 $c = \varepsilon(m)$ 以及 $\mathrm{m} = D(c)$。

第三章　同态加密

从广义上来说，同态加密技术可以被认为是一个加密系统，因为它除了允许用户可以对明文数据进行基本的加密操作和解密操作之外，还可以在加密得到的密文数据上进行相应的计算操作。如果某个加密系统对于密文只可以进行某些有限的计算操作，不能够进行全部的计算操作，那么将它称为部分同态技术。例如：对于使用未进行填充的非对称加密算法RSA的加密系统，只可以被认为是具有乘法的同态性。即，对于明文m_1和明文m_2，我们可以得到$\varepsilon(m_1)=m_1e\%N$和$\varepsilon(m_2)=m_2e\%N$，其中N是RSA的模数，e是RSA公钥的指数，因此，我们可以得到$\varepsilon(m_1)\varepsilon(m_2)=(m_1e\cdot m_2e)\%N=(m_1\cdot m_2)e\%N=\varepsilon(m_1\cdot m_2)$。通过该公式可以看出，RSA只允许我们对密文只能够进行简单的乘法操作而不能进行加法以及其他计算的操作。对于满足加法同态性的加密技术是Paillier加密算法，如果我们的公钥是模N并且是基于g的，那么对于明文信息m进行加密则可以表示为$\varepsilon(m)=g^m r^N\%N^2$，其中r是在区间[1,N-1]之间任意的一个数，Paillier加密算法的加法同态特性则可以表示为$\varepsilon(m_1)\varepsilon(m_2)=(gm_1r_1N)\cdot(gm_2r_2N)\%N_2=gm_1+m_2(r_1r_2)N\%N_2=\varepsilon(m_1+m_2)$。可以看到，Paillier密码算法是满足加法的同态特性，但是不满足乘法等其他操作的同态性。

从另一个方面来讲，完全同态加密系统是一种允许客户端除了可以进行基本的加密操作与解密操作之外，还可以让其在加密数据上进行任何的计算操作而不需要去解密数据。假设存在两个明文m_1和m_2，$c_1=\varepsilon(m_1)$和$c_2=\varepsilon(m_2)$分别是对相应的明文进行加密操作后得到的密文，根据完全同态加密的性质，我们可以得出$c_1+c_2=\varepsilon(m_1+m_2)$和$c_1\cdot c_2=\varepsilon(m_1\cdot m_2)$。可以看到，完全同态加密是可以对密文数据进行任意的计算操作。因此，用户可以将自己的数据同态同态加密技术加密，然后发送到远程的云服务器上，当需要对这些数据进行某些复杂的计算操作时，可以利用云环境中强大的计算能力对数据进行处理并且是直接在云服务器上对相应的密文进行操作，再将计算的结果以密文的形式发送给用户，最后用户自己解密得到预期的结果。

从数学上来说，一个完全同态加密算法可以被认为是由一个四元组的多项式组成$(Gen,Enc,Dec,Eval)$，验证：

$\mathrm{Gen}(\lambda)$：它是一种密钥生成的算法，通过输入的一个安全参数λ，然后输出公

共的和私密的密钥对(pk, sk)， pk 表示公钥， sk 表示私钥。

Enc(m, pk)：它是一种加密的算法，对于输入明文 m 和公钥 pk，通过该加密算法作用后，输出相应加密后的密文 c。

$Dec(c, sk)$：它是一种解密的算法，对于输入密文 c 和私钥 sk，通过该解密算法的作用后，我们可以得到相应的明文 m。

E$val(C, c_1, c_2, ..., c_n)$：它是一种对密码算法评估的算法，对于输入的环 C 和密文 $c_1, c_2, ..., c_n$，并且验证 D$ec(Eval(C, c_1, c_2, ..., c_n), sk) = C(m_1, m_2, ..., m_n)$，因为对于评估算法，任何人都可以使用评估，因次，它不需要相应的私密密钥 sk。

接下来，我们的方案将会基于环的加密系统，它是可以允许客户端对于输入的明文数据进行逐位加密得到相应密文。

假设我们有这样的一个数据库，它里面有许多私密的数据，这些数据通过 $c_1, c_2, ..., c_n$ 表示，并且我们的数据库是放在远程云服务器上的。

第四章　测试两个加密消息是否相等

对于采用完全同态加密的方案下，它可以允许使用者在不解密消息的条件下测试两个加密的信息相等或者不相等。在本章节中，我们将主要研究对于两个加密后的 c_1 和 c_2 是否相等，可以被表示为 $\mathrm{equal}(c_1, c_2)$。它可以接收两个输入的明文 c_1 和 c_2，它们有 $c_1 = \varepsilon(m_1)$ 和 $c_2 = \varepsilon(m_2)$，我们可以验证 m_1 和 m_2 是否相等但是不需要对其进行解密得到所给定的明文。为了实现这个操作，我们需要在密文上做以下两步操作。

5.1 反转

对加密数据的按位进行反转是一种可以允许我们改变一个位并且仅仅只是基于密文进行的反转操作，这种完全是可以通过完全同态加密算法来实现的，例如：

假设，对于 θ，我们有 $\theta \in \{0,1\}$，并且 $c = \varepsilon(\theta)$，我们可以得到 $c = \varepsilon(\bar{\theta})$，使得 $\bar{\theta} = \theta \oplus 1$，其中 $\bar{\theta}$ 是 θ 的反转。

c 是可以通过 c 的迭代计算得到，表示为 $c = c + \varepsilon(1) = \varepsilon(\theta) + \varepsilon(1) = \varepsilon(\theta + 1) = \varepsilon(\bar{\theta})$。

5.2 补码

二进制的补码是被定义为通过反转二进制中所有的位，然后对最后一位加 1 获得的值的一种数字表示形式，这个是一种同时反转多个位得到的数值。为了计算出逐位输入加密的补码，我们需要对每个加密位进行反转。

令 m 位二进制的明文（并不需要是一位）加密成 $c = \varepsilon(m)$ 并且 $c = \varepsilon(\overline{m})$，其中 $\overline{m}$ 是 m 的补码，我们可以得到 $c = \varepsilon(m) + \varepsilon(111...1) = \varepsilon(m \oplus 111...1) = \varepsilon(\overline{m})$。

5.3 相等

使用上述的方式判断两个明文消息是否相等是优于 $\mathrm{equal}(c_1, c_2)$ 直接以两个密文作为输入然后判断的方式，因为 equal 操作将会返回加密的结果，它之前是通过单个位来进行解密。如果位的值是 1，那么证明 m_1 和 m_2 是相等的，否则说明 m_1 和 m_2 是不相等的。

为了在只使用 c_1 和 c_2 的条件下测试 m_1 和 m_2 是否相等，我们需要执行以下操作：

1.先对密文 c_1 和 c_2 按位进行加法运算，表示为 $c_1 \oplus c_2 = \varepsilon(m_1 \oplus m_2)$。

如果 m_1=m_2，我们可以得到 $C = c_1 \oplus c_2 = \varepsilon(000...0)$，否则，我们将在 $c_1 \oplus c_2$ 运算的加密位中得到只有一个位等于 1。

2.计算 $c_1 \oplus c_2$ 的补码 $\overline{c_1 \oplus c_2}$ 。

3.将得到的结果的所有加密位进行乘法操作，我们将会得到一个被加密的位（0或1）。

4.如果我们最终得到的结果是 $equal(c_1, c_2) = \varepsilon(1)$，那么我们可以得出结论 $m_1 = m_2$，如果得到结果是 $\text{equal}(c_1, c_2) = \varepsilon(0)$，那么则证明 $m_1 \neq m_2$。

第五章　加密数据的搜索

当企业在选择是否使用云服务器来进行数据的存储以及应用程序的运行容器时，安全仍然是他们在使用云计算中考虑的最主要的部分。现如今，对明文数据进行加密处理得到密文，然后将数据以密文的形式存储的远程云服务器的数据库上是非常必要的。通过加密明文数据，可以使我们的隐私数据得到更多的保护，更加安全，但是通常情况下，这也意味着我们为了保证数据的安全性而不得不牺牲一定的功能性。对加密数据的搜索是一种很意义的解决方案，它允许云服务器可以任意的搜索用户的密文数据，同时会基于一个称为 trapdoor 的令牌，它包含了客户端传递的需要云服务器进行搜索的关键字。当然，客户端在传递那些被搜索的关键字时，它们也是必须需要加密的，相应这样也会增加一定的性能消耗。因为云服务器对用户密文数据的搜索是任意的，这也意味着云服务器不需要获取任何不必要的信息，它只是需要关于搜索的加密关键字和在整个过程中对加密数据的查询操作。

完全同态加密允许我们查询那些加密的数据而不需要对数据进行相应的解密操作，除了对密文的基本计算操作之外，我们还可以对密文进行任意的搜索。

假设，我们有这样的一个数据库，它里面有密文数据 $c_1, c_2, ..., c_n$，为了判断里面是否有明文 m 加密的密文信息，表示为 $c = \varepsilon(m)$，在将查询请求发送到远程云服务器之后，我们可以进行如下的操作，具体过程如下：

先从查询的请中获取相应加密的消息 c 并且判断它是否与我们数据库中加密数据的某一行记录相等。因为在这个阶段，我们可以获取到 n 个加密位，其中 $b_1 = equal(C, c_1)$，$b_2 = equal(C, c_2)$，. . .，$b_n = equal(C, c_n)$。

如果数据库中不存在加密消息 c，那么 n 个加密位都会等于被加密的 0，即表示为：$b_1 b_2 b_3 ... b_n = \varepsilon(000...0)$，如果数据库中存在加密消息 c，则我们将会获得至少一个位 $b_i = \varepsilon(1)$，而对于给定的 $i \in [1, n]$，会有 $b_1 b_2 b_3 ... b_n = \varepsilon(000...1...0)$。

因此，要验证数据库中是否存在加密的消息 c，远程的云服务器应该对 $(b_1 b_2 b_3 ... b_n)$ 的计算结果和 $\varepsilon(000...0)$ 进行判断它们是否相等。即通过计算 $SEARCH = equal(b_1 b_2 b_3, ..., b_n, \varepsilon(000...0))$。得到的结果将会被反转并且发送给请求方。即：SEARCH。

请求方在收到加密后的 SEARCH 之后，会对其进行解密。相应的将会展现出两种结果：如果他发现是得到的结果是 0，那么则说明通过密文查询的数据不存在，否则，他发现得到的结果是 1，这也就意味着通过密文查询的结果是存在的。

第六章　PIR

假设某个客户端已经查询了数据库并且得到一个肯定的结果即所搜索的加密数据是存在于数据库中的。此时，客户端想要获取确切的加密数据，可以通过使用 PIR 协议来获得，PIR 允许客户端获取它想要的信息并使用一种更加安全的方式而且不需要云服务器去决定哪些部分是需要被选中的。PIR 协议可以与对加密数据搜索的方案进行组合来使用，这样会得到一个更加安全的协议，它可以允许客户端在云上对加密数据进行相应的搜索并且进行私人信息的检索。如果存放在云上的数据是用户的隐私数据时，这种方案可以非常有效的保护用户数据隐私。完全同态加密能够允许对加密数据的搜索以及私人信息检索所需的数据库的数据。

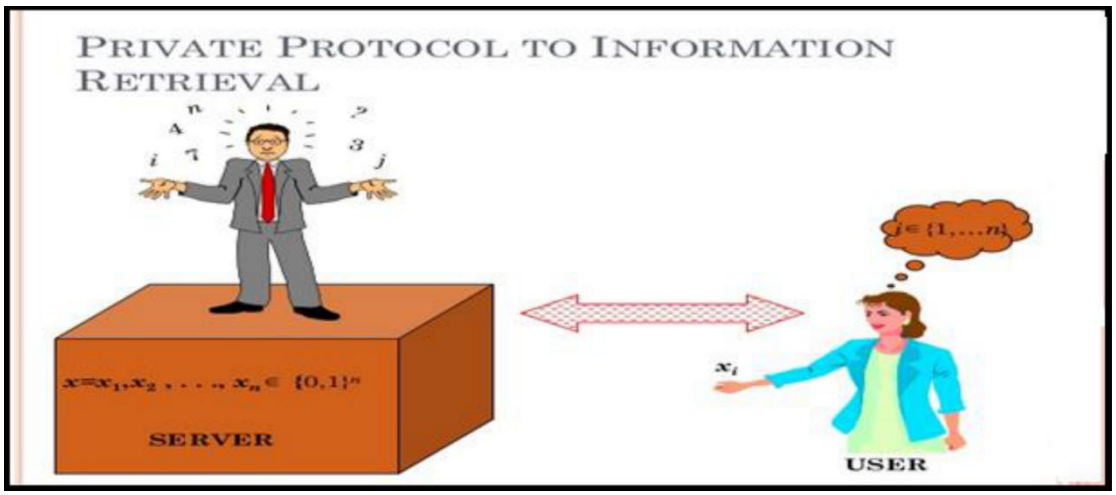

图 6-1 私人信息检索协议

6.1 完全同态加密中的 PIR

PIR 协议是可以通过完全同态加密技术实现的。假设，数据库中存在相应的数据 $e_1e_2e_3...e_n$，它们整体是通过一个独一无二的整数进行组织起来的，并且这些数据在数据库中是不会被加密的，它们只是简单的整数。假设我们有一个完全同态的加密方案，它允许我们在不需要知道数据库的所有者的前提下获取到第 k 条记录。

客户端首先通过明文计算出相应加密后的密文 c_i ，其中 c_i 是除 $i = k$ 之外的对 0 的加密，在这种情况下 ck 是对 1 的加密。因为我们的密码系统是一种基于概率的解决方案，因此 0 是可以以不同的方式被加密，并且所有的 c_i 也都是不相同的。客户

端将需要查询的密文数据$(c_1c_2c_3....c_n)$发送到远程的云服务器上，数据库的管理员计算$c_1e_1+c_2e_2+c_3e_3+...+c_ne_n$的结果，并将结果返回给客户端。然后，他再计算$D(c_1e_1+c_2e_2+c_3e_3+...+c_ne_n)$的结果并且对$e_k$进行检索。最后我们将有$c_1e_1+c_2e_2+c_3e_3+...+c_ne_n=\varepsilon(0)e_1+\varepsilon(0)e_2+\varepsilon(1)e_k+...+\varepsilon(0)e_n=\varepsilon(0*e_1+0*e_2+...+1*e_k+...+0*e_n=\varepsilon(e_k)$。

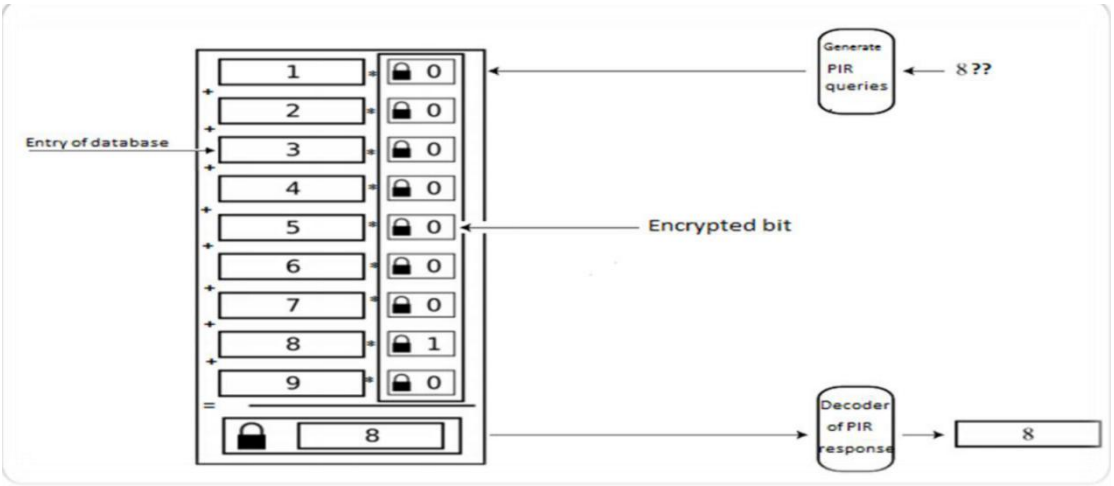

图 6-2 全同态加密中的 PIR 协议

上图展示了如何进行私人信息检索并且获取的，先是用户将需要检索的数据加密，上图中查询的是 id 为 8 的记录，然后请求数据库进行相应的查询，数据库中存放的是密文的记录，当在数据库中匹配到 id 为 8 的数据后，将该密文记录记录发送给用户，用户获得数据后解密得到相应的明文数据。

结　论

在本论文中，我们主要解决了对加密数据的搜索和私人信息检索（PIR）协议等问题，其中所有的操作都是在存储数据的远程服务器中进行的，在数据库数据加密的情况下，对于在云环境中的数据搜索，将对加密数据搜索与私人信息检索放在一个协议里是一种非常高效的解决方案，我们将这两种技术进行组合的前提都是它们基于完全同态加密技术的。所有的操作都简单的在同一个加密系统中执行的，它允许我们在加密数据上对密文进行搜索而不用事先对其进行解密，从而不会暴漏相应的明文信息，保证了用户隐私数据的安全。

参考文献

[1] D. Song , D. Wagner et A. Perrig, «Practical Techniques for Searches on Encrypted Data,» IEEE Symposium on Security and Privacy, pp. 44-55, 2000.

[2] D. Boneh, G. Crescenzo, R. Ostrovsky et G. Persiano, «Public key encryption with keyword search,» chez EUROCRYPT (LNCS), 2004.

[3] B. Chor, O. Goldreich, . E. Kushilevitz et M. Suda, «Private information retrieval,» In Proc. of the 36th Annu. IEEE Symp. on Foundations of Computer Science Pages 41–51, 1995. Journal version: J. of the ACM, 45:965–981, 1998.

[4] R. Rivest, L. Adleman et M. Dertouzos, ««On Data Banks and Privacy Homorphisms,»,» In Foundataions of Secure Computataion, Academic Press, pp. 169-179, 1978.

[5] C.Gentry, «A fully homomorphic encryption scheme,»https://crypto.stanford.edu/craig/craigthesis .pdf, September 2009.

[6] M. van Dijk, C. Gentry, S. Halevi et V. Vaikuntanan, «« Fully homomorphic encryption over the integers,»,» Cryptology ePrint Archive, Report 2009/616, 2009. http://eprint.iacr.org/..

[7] Z. Brakerski, C. Gentry et V. Vaikantanathan, «« Fully Homomorphique Encryption without Bootstrapping,»,» Available at http://eprint.iacr.org/2011/277..

[8] J. Fan et F. Vercauteren, ««Somewhat Practical Fully Homomorphic Encryption,»,» Available at http://eprint.iacr.org/2012/144. .

[9] J. Kim, M. Sung Lee, A. Yun et J. Hee Cheon, «“ CRT-based Fully Homomorphic Encryption over the Integers”,» Available at http://eprint.iacr.org/2013/057.

[10] A. Kipnis and E. Hibshoosh, « « Efficient Methods for Practical Fully Homomorphic Symmetric-key Encrypton, Randomization and Verification »,,» Cryptology ePrint Archive, Report 2012/637..

[11] R. Rivest, A. Shamir et L. Adleman, «A Method for Obtaning Digital Signatures and Public Key Cryptosystems,» Commununications of ACM, Vol. 21, pp. 120-126, April 1978..

[12] P. Paillier, «Public-key cryptosystems based on composite degree residuosity classes.,» chez 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic, 1999.

[13] D. Boneh, E. Goh et K. Nissim, «Evaluating 2-DNF formulas on ciphertexts. In Kilian, J., ed.: Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings. Volume 3378 of Lecture,» chez Theory of Cryptography, Second Theory of Cryptography Conference, Combridge, February 10-12, 2005.