International Journal of Security and Its Applications
Vol. 9, No. 6 (2015), pp. 61-70
http://dx.doi.org/10.14257/ijsia.2015.9.6.07

Hybrid Homomorphic Encryption Method for Protecting the Privacy of Banking Data in the Cloud

Maha Tebaa[1], Karim Zkik[2] and Said El Hajji[3]

[1, 2, 3]*University of Mohammed-V, Faculty of Sciences - Rabat, Morocco,*
Laboratory of Mathematics, Computing and Applications
[1]*maha.tebaa@gmail.com,* [2]*karim.zkik@gmail.com,* [3]*elhajji@fsr.ac.ma*

Abstract

The time and cost saving for banks when implementing cloud computing strategies are staggering. However, it is important to consider the security and protection of data when it comes to the widespread adoption of cloud. Fully homomorphic encryption is currently still undergoing experimentations. One of its limitation is the time required to encrypt and decrypt the sensitive data, as the traditional encryption systems showed a level of resistance and considerable maturity that can be rehabilitated or hybridized for application in the field of sensitive data protection hosted in the Cloud.

In this paper we will propose hybrid homomorphic system that will be applied to the banking data to perform operations on encrypted data without decrypting, based on the encrypting, the decryption and the operation treatment time on the ciphertext which were obtained by simulating an addition and multiplication homomorphic cryptosystem and comparing it with DGHV that is somewhat homomorphic; then we will choose our model which is most suitable for banking application.

Keywords: *Homomorphic Encryption, Paillier, RSA, Cloud Computing, bank data*

1. Introduction

Companies do not longer refute the need of storing its sensitive data on the cloud, which they already do using traditional encryption systems. The question here is how to process this data without decrypting it in order to respect the company's privacy. Hence the idea of using homomorphic encryption for banking data which is a new approach that can help banks take their data management to the highest level: the cloud [1].

Cloud Computing is not only a new seductive marketing trend, but it is an economic reality to access many services online. This comprehensive access offer to computing resources allows evolutionarily companies to focus on their core competencies by exempting them from their servers' management. The major question that keeps many companies reluctant regarding this new storing technology, including banks, is the security of their sensitive data hosted on the Cloud.

Cloud computing may offer several advantages in terms of cost reduction and scalability but raises new security issues regarding sensitive data.

Following this, and in order to perform operations on encrypted banking data stored in the cloud without decrypting them, we will opt for the homomorphic encryption [2, 3]. This latter consists in encrypting the client's identity with a multiplicative homomorphic encryption which corresponds to the "logical and" used for keyword searching into an encrypted database. Add to this, the data concerning a client's deposits of all his or her accounts is encrypted by an additive homomorphic encryption; aiming to compute the total amount of the client's deposits by the cloud server and deciphering the result at the level of client's private bank. In both cases, the result is the same, as if the calculations were made using raw data.

ISSN: 1738-9976 IJSIA

In Section 2, we will talk about the lack of Cloud adoption by banks. In Section 3, we will define Homomorphic Encryption and we will illustrate some examples of existing Homomorphic cryptosystems. In Section 4, the focus will be on our simulation and its analysis, in Section 5; we will present the scheme of our banking application with some examples of calculations on banking accounts of customers. The conclusion and perspectives are dealt with in Section 5.

2. Banking in the Cloud

The lack of cloud services usage, not only by banks but also by all the companies in all the fields, creates a strong incentive for the researches and cloud providers to adapt their solutions to the security required by banks, encouraging them to migrate and to outsource their computations to the cloud; allowing them to focus only on their core competencies.

The migration to the cloud will lead to a control loss over the customer bank data (account numbers and sum of deposits, loans allocations...).

There are many compelling reasons to migrate applications and data to a private or a public cloud: scalability, agility, cost savings and so on and so forth. However, those benefits come along with a new risk regarding the security of critical business data; a risk that has to be managed by any organization whose data migrated to the cloud using a robust solution for data encryption and encryption key management. This risk is illustrated through the USA PATRIOT Act [4]; an act that makes it possible for competitors and governments to access data through cloud providers without the consent of the data owner. Even storing data in foreign jurisdictions does not provide a sufficient guaranty that customers' data is out of reach of disclosure requirements.

To solve this problem, some security vendors require the installation of a physical key management server in the data center as a prerequisite. Other security providers require you to "trust" them and use their key management service. This approach violates the principle and the respect of the requirements keeping the keys under your own control.

The banking sector is renowned for the volume and speed of data it produces, transports and stores. Given the recent financial crisis and subsequent regulatory pressures, the cost of computers has grown exponentially not only for the banking organization as a whole, but even at the level of the division.

Nowadays, Banks have to cut their IT costs, but it should not be done at the stake of compromising their data security and integrity. The need, therefore, to explore the unconventional and innovative options reduced IT costs while maintaining data integrity. However, with limited resources and even limited budgets, a methodology for traditional IT implementation is neither effective nor desired [5].

The Cloud Computing [6], as a technology option for data and business processes, took credible foothold in recent times in several industry sectors including financial services. This paper details the "must knows" to implement an efficient and secure information management framework in a cloud environment. We will, first, explore the various delivery and deployment of cloud models available to a bank.

The security must be your cloud provider's top priority. As matter of fact, the highest portion of Amazon Web Services (AWS) investments is dedicated to increase this level of security is based on a model of shared responsibility: AWS [7] ensures the security of the lower layers (physical buildings, networks, virtualization...), and you are responsible for the security of your applications, OS and data on which you have total control (*e.g.*, administrator access to VM).

This means that you apply the same rules of security on your applications in the cloud infrastructure.

3. Homomorphic Cryptosystems

Cloud Encryption is the only solution that offers the convenience of encryption and key management in virtualized environments, without giving up data confidentiality.

Security requirements for data and algorithms have become stricter throughout the latest years. Due to the large growth of the technology, a wide variety of attacks on digital assets and technical devices are activated. For storage and safe data reading, there are several possibilities such as data encryption. The problem becomes more complex when asking to perform computation on encrypted data. This is where the homomorphic cryptographic systems can be used.

The partial homomorphic encryption allows some operations on encrypted data without decrypting it, examples: RSA, Paillier.

3.1. Multiplicative Homomorphic Encryption

The multiplicative homomorphic encryption

$$D_k(E_k(n) \times E_k(m)) = n \times m \quad \text{OU} \quad E_k(x \otimes y) = E_k(x) \otimes E_k(y)$$

Example of Multiplicative Homomorphic Encryption (RSA) [8]:

RSA : Multiplicative homomorphic encryption

Let $n = pq$, such that p and q are relatively prime, a and b are selected such that $ab \equiv 1 \pmod{\Phi(n)}$.
n and b are public; p, q and are private.

$$E_k(x) = x^b \bmod n$$

$$D_k(y) = y^a \bmod n$$

$E_k(x_1) \times E_k(x_2) = x_1^b x_2^b \bmod n = (x_1x_2)^b \bmod n = E_k(x_1x_2)$

3.2. Additive Homomorphic Encryption

The additive homomorphic encryption

$$D_L(E_L(n) \times E_L(m)) = n + m \quad \text{OU} \quad E_k(x \oplus y) = E_k(x) \otimes E_k(y)$$

Example of Additive Homomorphic Encryption (Paillier) [9]:

Paillier : Additive homomorphic encryption

Let p and q be two large prime numbers, et $n = pq$. λ denotes the Carmichael function as, $\lambda(n) = gcm(p-1, q-1)$.
Let $g \in \mathbb{Z}_n^*$ a random as $L(g^\lambda \bmod n^2)$ is invertible modulo n (où $L(u) = \frac{u-1}{u}$). n and g are public; p et q (or λ) are private.

$$E_k(x, r) = g^x r^n \bmod n^2$$

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

$E_k(x_1,r_1) \times E_k(x_2,r_2) = g^{x1+x2}(r_1r_2) \bmod n^2 = E_k(x_1 + x_2, r_1r_2)$

3.3. Somewhat Homomorphic Encryption (DGHV) [10]

DGHV : Somewhat homomorphic encryption

Encryption settings: r, p et q, $r \sim 2^n, p \sim 2^{n^2}, q \sim 2^{n^5}$, p prime, p is the private key.
$E_k(x) = pq + 2r + x$
$D_k(y) = (y \bmod p) \bmod 2$

Proof : pq is greater than 2r+x
thus y mod p = 2r + x
at last (y mod p) mod 2 = (2r + x) mod 2 = x

For two ciphertexts c_1 et c_2 :
$c_1 + c_2 = (q_1 + q_2)p + 2(r_1 + r_2) + m_1 + m_2$
So, if : $2(r_1 + r_2) + m_1 + m_2 << p$
Then $(c_1 + c_2) \bmod p = 2(r_1 + r_2) + m_1 + m_2$
so, it is homomorphic for addition
$c_1 \times c_2 = [q_1q_2p + (2r_1 + m_1) + (2r_2 + m_2)]p + 2(2r_1r_2 + r_1m_1 + r_2m_1) + m_1m_2$
So, if: $2(2r_1r_2 + r_1m_1 + r_2m_1) + m_1m_2 << p$
Then $(c_1 \times c_2) \bmod p = 2(2r_1r_2 + r_1m_1 + r_2m_1) + m_1m_2$
so, it is homomorphic for multiplication

4. Simulation and Analysis

Practical evaluation is performed in a MacBook Pro (MacBookPro 9,2), with the following characteristics:
Intel Core i5 @ 2000 MHz 4 GB - DDR3 @ 1600 MHz
Intel HD Graphics 4000

4.1. Result of RSA Simulation:

For RSA, the Following Table Shows the Different Simulation Values

Table 1. Simulation Results of RSA Multiplicative Homomorphic Encryption

RSA		Key size											
		256			512			1024			2048		
		Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time
Text size	1KB	58ms	3ms	61ms	331ms	4ms	326ms	2384ms	11ms	2376ms	17369ms	27ms	17668ms
	10KB	534ms	17ms	537ms	3284ms	29ms	3396ms	23607ms	77ms	23435ms	176848ms	234ms	176348ms
	100KB	40607ms	976ms	40676ms	260584ms	2332ms	284736ms	.....	.....	.....	.....	.....	.....
		Treatement duration											

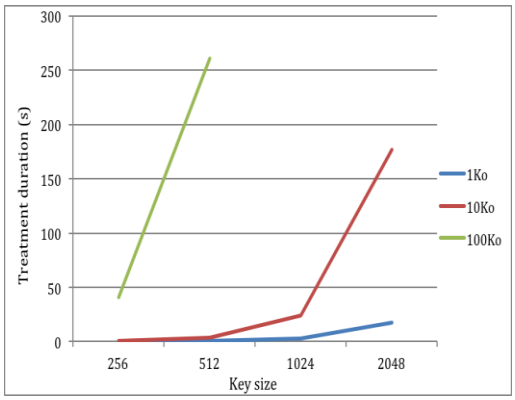


Figure 1. The RSA Encryption Time

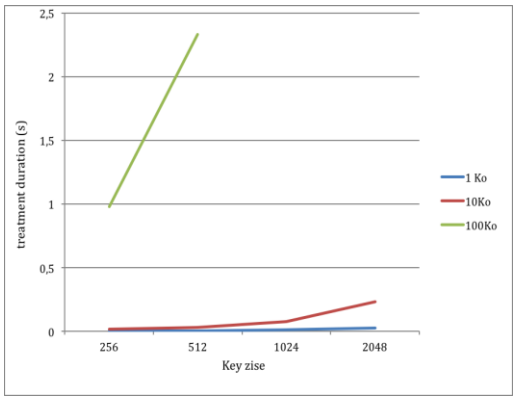


Figure 2. The Multiplication Time of Two Ciphertexts (C1xC2)

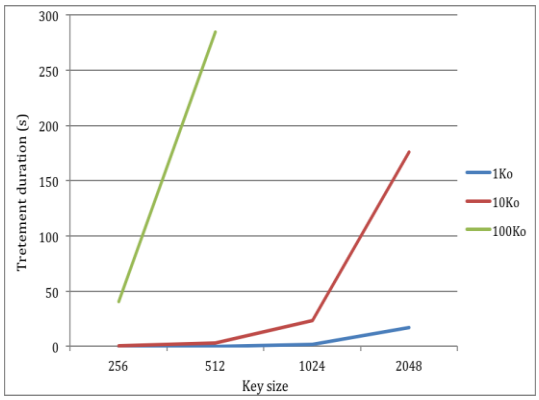


Figure 3. Decryption Time Dec (C1xC2)

- For 1KB text size, RSA homomorphic cryptosystem is able to encrypt, multiply two cleartexts and decipher the result of the multiplication using a 2048 key size.
- For 10KB text size, RSA homomorphic cryptosystem is able to encrypt, multiply two cleartexts and decipher the result of the multiplication using a 2048 key size, but with a very elevated processing time.
- For a 1024 key size the RSA homomorphic cryptosystem is no longer able to multiply two ciphertexts of 100KB each, so it is limited in 1024 key size and 100KB cleartext size.

4.2. Result of Paillier Simulation:

For Paillier, The following table shows the different simulation values

Table 2. Simulation Results of Paillier Additive Homomorphic Encryption

Paillier		Key size											
		256			512			1024			2048		
		Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time
Text size	1KB	1346ms	25ms	2761ms	8671ms	71ms	17010ms	63062ms	175ms	126209ms	483979ms	526ms	971372ms
	10KB	11438ms	214ms	21523ms	79584ms	569ms	162244ms	624020ms	1798ms	1186745ms	1395256ms	2265ms	1404403ms
	100KB	117612ms	2191ms	240827ms	.....	.....	.....	.....	.....	.....	.....	.....	.....
		Treatement duration											

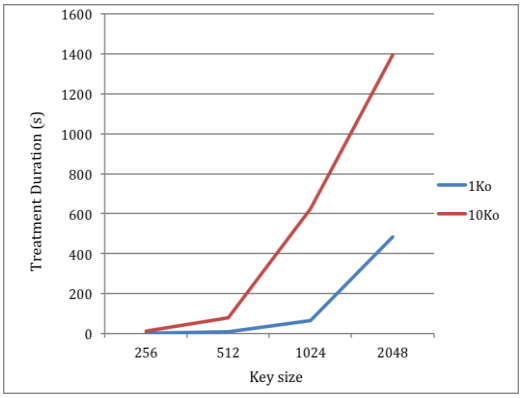


Figure 4. The Paillier Encryption Time

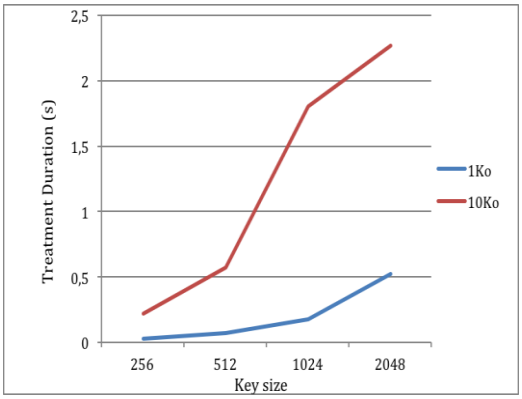


Figure 5. The Multiplication Time of Two Ciphertexts (C1xC2)

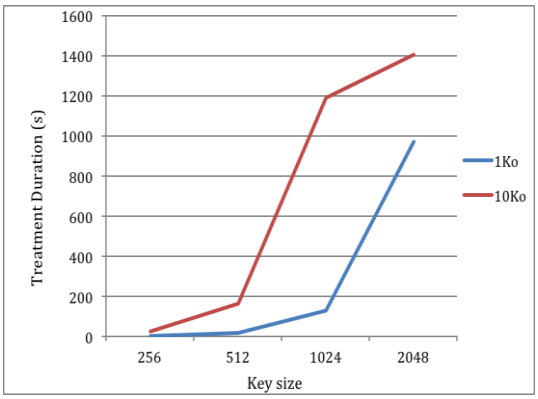


Figure 6. Decryption Time Dec (C1xC2)

- For 1KB text size, Paillier homomorphic cryptosystem is able to encrypt, multiply two cleartexts and decipher the result of the multiplication using a 2048 key size, with a processing time of 16.18 minutes.
- For 10KB text size, Paillier homomorphic cryptosystem is able to encrypt, multiply two cleartexts and deciphering the result of the multiplication using a 1048 key size, but with 19.77 minutes for deciphering, and 23 minutes to decipher the result using 2048 key size.
- For a 512 key size the Paillier homomorphic cryptosystem is no longer able to multiply two ciphertexts of 100KB each, so it is limited at 512 key size and 100KB of cleartext.

4.3. Result of DGHV Simulation:

For DGHV, the following tables show the different simulation values

Table 3. Simulation Results of DGHV Additive Homomorphic Encryption

DGHV (Addition)		Key size					
		64			128		
		Encryption time	(C1+C2) time	Dec(C1+C2) time	Encryption time	(C1+C2) time	Dec(C1+C2) time
Text zise	128 bytes	44165ms	8527ms	2688ms	3081206ms	--------	--------
	256 bytes	53723ms	19836ms	6256ms	--------	--------	--------
		Treatement duration					

Table 4. Simulation Results of DGHV Multiplicative Homomorphic Encryption

DGHV (Multiplication)		Key size					
		64			128		
		Encryption time	(C1xC2) time	Dec(C1xC2) time	Encryption time	(C1xC2) time	Dec(C1xC2) time
Text zise	128 bytes	46112ms	15364ms	10401ms	2855851ms	--------	--------
	256 bytes	55254ms	--------	--------	--------	--------	--------
		Treatement duration					

- For 10KB text size, DGHV additive homomorphic cryptosystem is able to encrypt, multiply two cleartexts and deciphering the result of the multiplication using a 1048 key size, but with 19.77 minutes for deciphering, and 23 minutes to decipher the result using 2048 key size.
- For a 512 key size the DGHV multiplicative homomorphic cryptosystem is no longer able to multiply two ciphertexts of 100KB each, so it is limited at 512 key size and 100KB of cleartext.

5. Hybrid Homomorphic Encryption Applied to Banking Data in the Cloud

5.1. Application' Scheme

Our application scheme is as follows:

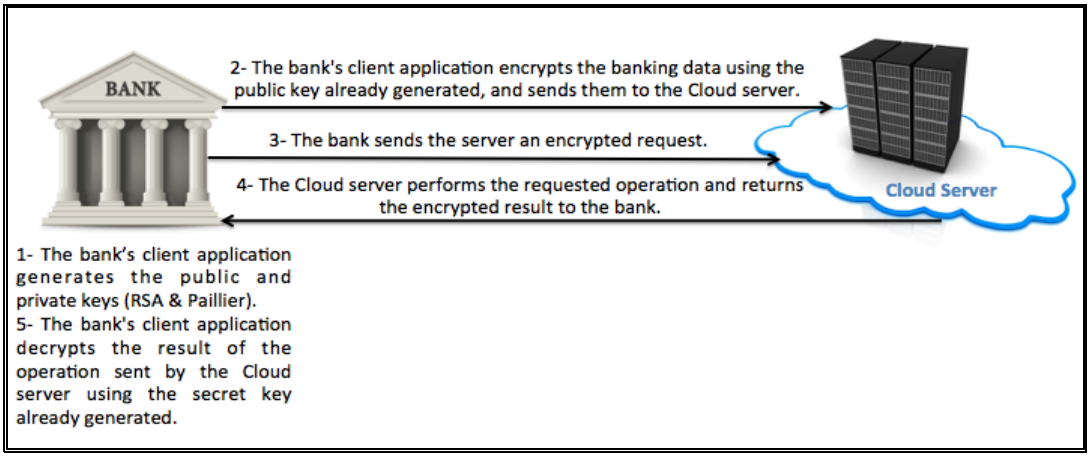


Figure 7. Scheme Representing the Link between a Client Company (Bank) and its Data Hosted in a Cloud Provider Server

VPN network linking the customer company (bank) with the server in the cloud;
Our Cloud platform is set on a Citrix [11] server.

1. The bank's client application generates encryption keys with Paillier and RSA cryptosystems, encrypts data, and transmits it to the cloud server
2. The names and account numbers of customers are encrypted with RSA to enable keyword search in the encrypted data without decrypting it.
3. The sum of deposits for each customer's account is encrypted with Paillier to allow an addition operation on all the accounts of the same customer.
4. The bank sends a request to calculate the total amount of the customer's deposits
5. The Cloud server performs the calculations on encrypted data and return the encrypted result to the bank

6. Using the private key, the client application decrypts the encrypted result and obtains the same sum as if the calculation was carried out on the raw data.

5.2. Parameters and Results

For our banking application, the data to be encrypted does not exceed 1KB (Name, Account number and amount of money). So we chose to execute a keyword search processing on an encrypted database and compute the sum of a customer deposits, whose data has been encrypted and stored in the cloud using a 1024 key size for RSA that requires 2,384s to encrypt, 11ms to calculate the product of two ciphertexts and 2,37s for deciphering.

For Paillier homomorphic cryptosystem we chose a key size 512 that requires 8,671s to encrypt, 71ms to compute the sum of two ciphertexts and 17,01s for deciphering,

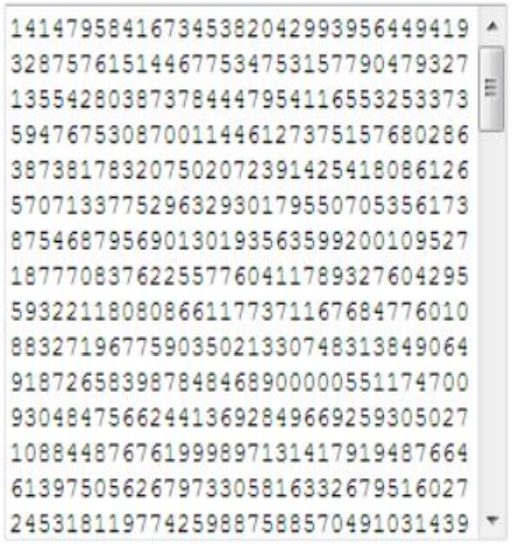
At the level of the bank application, we have this database:

Account-ID	Last Name	First Name	Account Number	Amount Money
2	Tebaa	Maha	1230001	10000 MAD
3	Tebaa	Maha	1230002	200500 MAD
4	Zkik	Karim	1250001	30000 MAD
5	Zkik	Karim	1250002	10500 MAD
6	El Hajji	Said	1240001	20000 MAD
7	El Hajji	Said	1240002	40000 MAD

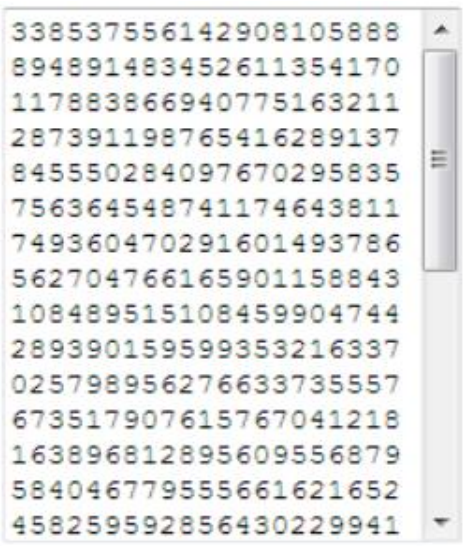
At the level of the Cloud server, data is stored encrypted:

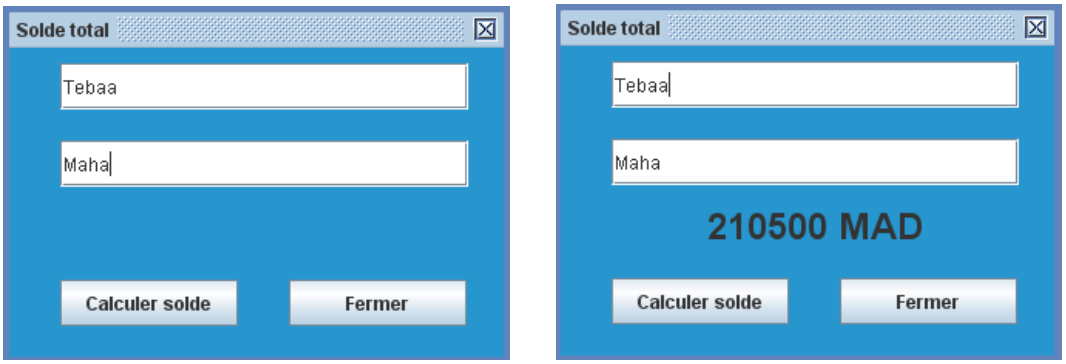
2	1414795841673453	0067857878412112	15951989389425504	338537556142908105888894891483452611354170
3	1414795841673453	0067857878412112	15951989389425504	167724147567873038728945228
4	1341507679240820	0763165154812574	15951989389425504	366000508608163319867483949
5	1341507679240820	0763165154812574	15951989389425504	200375752527645824480601627
6	0630263810432476	1807227423402023	15951989389425504	818644300135799317042869064
7	0630263810432476	1807227423402023	15951989389425504	130659140102188133189693555

The customer's name encrypted using RSA



The customer account balance encrypted using Paillier





Indeed, the Balance of all the Accounts Hold by the Customer “Tebaa Maha” is 10000MAD+200500MAD= 210500 MAD

5. Conclusion

This hybrid system to respect the confidentiality and privacy of the database of bank accounts stored in the cloud will encourage cloud providers to integrate the homomorphic encryption in the deployment of their security solutions, to encourage banks to enjoy the benefits cloud computing [12], among the problems that exist is the management of private keys generated at the bank, you have to predict a very elevated level of protection and have a backup system to store the encryption keys and deciphering, because it is the only information that allows to access the encrypted data stored in the cloud.

References

[1] M. Tebaa and S. El Hajji, “From Single to Multi-clouds Computing Privacy and Fault Tolerance”, IERI Procedia, Elsevier, vol. 10, **(2014)**, pp. 112-118.
[2] M. Tebaa and S. El Hajji, “Secure Cloud Computing through Homomorphic Encryption”, International Journal of Advancements in Computing Technology, (IJACT), vol. 5, **(2013)**.
[3] L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen and A. Vasilakos, “Security and privacy for storage and computation in cloud computing”, Information Sciences, vol. 258, **(2014)**, pp. 371-386.
[4] M. Parashar, M. Abdel Baky, I. Rodero and A. Devarakonda, “Cloud paradigms and practices for computational and data-enabled science and engineering”, Computing in Science & Engineering, vol. 15, no. 4, **(2013)**, pp. 10-18.
[5] S. Pearson, Y. Shen and M. Mowbray, “A privacy manager for cloud computing”, Cloud Computing, Springer Berlin Heidelberg, **(2009)**, pp. 90-106.
[6] P. Manish, A. Moustafa, R. Ivan and D. Aditya, “Cloud paradigms and practices for computational and data-enabled science and engineering”, Computing in Science & Engineering, IEEE, vol. 15, no 4, **(2013)**, pp. 10-18.
[7] AWS: http://www.cloudmagazine.fr/avis-expert/je-construis-mon-datacenter-dans-le-cloud-en-5-minutes -top-chrono.
[8] R. Rivest, A. Shamir and L. Adleman, “A method for obtaining digital signatures and public key cryptosystems”. Communications of the ACM, 120-126, 1978. Computer Science, Springer, **(1999)**, pp. 223-238.
[9] P. Paillier. “Public-key cryptosystems based on composite degree residuosity classes”, 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic, vol. 1592, **(1999)**.
[10] M. Van Dijk, C. Gentry, S. Halevi and V. Vaikuntanathan, “Fully homomorphic encryption over the integers”, Advances in cryptology- EUROCRYPT 2010, Lecture Notes in Computer Science, Springer, vol. 6110, **(2010)**, pp. 24–43.
[11] R. van der Linden, D. Halls, S. Waterhouse and P. Benoit, “Systems and methods for establishing a cloud bridge between virtual storage resources U.S. Patent No. 8,578,076”, Washington, DC: U.S. Patent and Trademark Office, **(2013)**.
[12] D. Chen and H. Zhao, “Data security and privacy protection issues in cloud computing”, Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference, IEEE, vol. 1, **(2012)**, pp. 647-651.