Database Secure Manipulation based on Paillier's Homomorphic Encryption (DSM-PHE)

https://doi.org/10.3991/ijim.v13i12.11396

Somchai Prakancharoen
King Mongkut's University of Technology North Bangkok, Bangsue Bangkok, Thailand
somchai-prakan@hotmail.com

Abstract—The objective of this research was to suggest some simple solution to increase database manipulation security. Based on advantage of Homomorphic encryption, user's data in database is always encrypted by Homomorphic encryption algorithm such as Paillier. User's data is manipulated or processed on many times such as addition, subtraction, multiplication and division. By the advantage of Homomorphic encryption algorithm, user's data will not be decrypted while manipulated. Therefore, secrecy of data is still kept. Unfortunately, Paillier's Homomorphic encryption normally covers only addition and multiplication operation on numerical data. This paper suggests simple technique to enhance Pailler's encryption algorithm to perform data operation as well in subtraction and division operation. Evaluation in suggested database manipulation, DSM-PHE, indicates that its operation tasks take five operation times more than ordinary data base manipulation operation tasks, without any encryption. Therefore DSM-PHE shall be especially used in more sensitive data.

Keywords—Database manipulation, homomorphic encryption, paillier's encryption algorithm

1 Introduction

In database manipulation, there are many problems in secrecy or privacy of kept data in database engine. User's plain data which are kept in database may be deliberated attacked by both internal and external intruders.

Some database's administrator try to protect data records by encrypting all of these data while they arein database engine. Unfortunately, inquired data for data manipulation event have to be revealed especially when data processing is on performing. This disclosure user's data wasobserved or even copied by data manipulation officer and/or others.

This research suggested a solution to conceal user's data all the time while they are kept in database engine and even on data processing by anyone that performs a task by using user's data.

Paillier's Homomorphic encryption algorithm is used to solve this problem. Data are always protected as long as user's data are not need to be revealed by authorized actor.

Paillier's encryption is an algorithm of encryption that possesses Homomorphic property. The latter covers addition and multiplication operation. Unfortunately, subtraction and division operation are not in the operation.

This paper presented a simple technique to enhance Paillier's encryption to handle subtraction and division operation.

2 Related Theory and Research

2.1 Homomorphic encryption scheme[1]

Homomorphic encryption is a form of confidential encryption system based on homomorphism property in abstract algebra. Encrypted data or cipher text could be mathematical computed. It should provide a result of operation in the same value as identical plain text computation. There are many encryption algorithms that conform with Homomorphic property such as RSA, Pallier, etc.

Unfortunately, many traditional encryption algorithms do not cover all mathematical computation. For example, RSA encryption is covered just a multiplication homomorphic property. While, Paillier's encryption algorithm has multiplication and addition Homomorphic property.

This paper chose Paillier's encryption algorithm because it can easily enhance both subtractions and division mathematical operations. Therefore, all basic computations (+, -, * and /) are all operated on encrypt data like performing operation on plain data.

2.2 Paillier's encryption algorithm[2]

Paillier's encryption algorithm has addition and multiplication Homomorphic property.

Homomorphic property of Paillier's encryption algorithm

Addition

Additive Homomorphic property of Paillier's encryption is shown in equation (1).

$$D(E(m1, r1).E(m2, r2) \bmod n^2) = m1 + m2 \bmod n \quad (1)$$

Denote: D (decryption), E (encryption), m1(1st plain text), m2(2nd plain text) and n is devisor of Modulo operation. "r1" and "r2" are arbitrarily random number chosen by one who conducts encryption.

Multiplication

Multiplication Homomorphic property of Paillier's encryption is shown in both equations (2) and (3).

$$D(E(m1, r1)^{m2} \bmod n^2) = m1.m2 \bmod n \quad (2)$$

$$D(E(m2,r2)^{m1} \bmod n^2) = m1.m2 \bmod n \tag{3}$$

Overview of Paillier's encryption algorithm

Key generation

Step 1: Cryptographer, Database administratorDA: arbitrarily assign two prime numbers (p, q) that greatest common divisor, gcd, of p.q and (p-1).(q-1) must follow property in equation (4).

$$gcd(p.q,(p-1).(q-1)) = 1 \tag{4}$$

Step 2: calculate for n from equation (5).

$$n = p.q \tag{5}$$

Step 3: calculate for λ from equation (6).

$$\lambda = lcm((p-1).(q-1)) \tag{6}$$

Note that lcm is least common multiplication.

Step 4: integer positive number, call g, is arbitrarily assigned. This value must be ranged in $g \in Z^*_{n^2}$ andfollow property in equation (7).

$$gcd(g,n^2) = 1 \tag{7}$$

Step 5: calculate for value of μ from equation (8).

$$\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n\,; L = \frac{U-1}{n} \tag{8}$$

While U is data embed in function $L(U)$;$U = g^{\lambda} \bmod n^2$

Step 6: Therefore, public key is (n,g) and private key is (λ,μ).

If "p" and "q" have equivalent character size then variables setting should be performed as following.

$\phi(n) = (p-1).(q-1)$,$\mu = \phi(n)^{-1} \bmod n$,$\lambda = \phi(n)$,$n = p.q$ and $g = n+1$.

Data Encryption

If "m" is a plain text and "c" is cipher text. Encryption "m" under public key is shown in equation (9).

$$c = g^m.r^n \bmod n^2 \tag{9}$$

While, "r",$r \in Z^*_{n^2}$ and $0 < r < n$, is arbitrarily chosen by one who undertake plain data encryption. It must be positive integer that follow$gcd(r,n) = 1$criteria. Ordinary, c-cipher text value is an element in range;$c \in Z^*_{n^2}$.

Data Decryption

Plain text, "m", is revealed by decrypt cipher text with private key as shown in equation (10).

$$m = L(c^{\lambda} \bmod n^2).\mu. \bmod n \tag{10}$$

Modular data inversion

In order to overcome the Paillier's encryption algorithm properties that can't support subtraction and division mathematical operation, data inversion technique was used to transform data to itsinverse form. This inversed data can be performed by two ordinary support operation. Therefore, these strategies can enhance Paillier's encryption to support subtraction on addition and division on multiplication.

Modular additive inversion [3]

Modular additive inversion technique is used to overcome problem of Paillier's encryption system that is not support subtraction operation. If "y" is and operand that should be performed "x-y" mathematical operation with operand "x". "v", modular additive inverse value, is substituted "y" value. Using this additive inverse value, subtraction operation is then reversed to addition operation.

Therefore, subtract operation,$x - y \bmod n$, should be equivalence to $x + v \bmod n$. Value of "v" is calculated from $y + v \bmod n = 0$ while "0" is an identity value of addition operation.

For example, let v is modular additive inversion of 4 mod 7 then v can be calculated by following equation.

$$4 + v \equiv 0 \bmod 7$$

$$v \equiv -4 \bmod 7 \equiv -4 + 1.(3 + 4) \bmod 7 \equiv 3 \bmod 7$$

Hence, 3 is modular additive inversion of 4 mod 7.

Modular Multiplicative Inverse [4]

In order to perform mathematical, "x/y", operation, operand "y" must be changed to a modular multiplicative inversed value, "v". "v" is an inverse value form of "y" that can be calculated from $v = y^{-1} \bmod n$.

Therefore, division operation, $\frac{x}{y} \bmod n$, is then transformed as equation (11).

$$\frac{x}{y} \bmod n = x.y^{-1} \bmod n = x.v \bmod n \tag{11}$$

Let v is modular multiplicative inversion of "v mod m". Check for gcd if its value is "1"$gcd(y, m) = 1$. If "gcd" criteria is achieved then modular multiplicative inversion is feasible calculation. If there are exist any integer "a" and "b" that follow $a.m + b.y = 1$ criteria then "b" is a modular multiplicative inversion of "y".

For example, let v is modular multiplicative inversion of 4 mod 11. Check greatest common divisor of "y" and "m". Result of "$gcd(4,11) = 1$" is passes criteria.

Then "$a.11 + b.4 = 1$" is assigned. There exist two numbers, "a=-1" and "b=3", so that $a.11 + b.4 = -1.(11) + 3.(4) = -11 + 12 = 1$. Hence "3" is a modular multiplicative inversion of 4 mod 11.

Exclusive .or. (.xor.) Operation [5]

Exclusive or, .xor., is a logical mathematical operation that result is "true" value if either operand is "true" and both operands are not "true" or "false".

For example, if operand "A" value is "1001" and "B" value is "0101" then "A" .xor. "B", "C", value is "1100". If "C" value is .xor. with operand "B" value again then "B" .xor. "C" value is "1001" which equal to "A" value. That is, perform "A" .xor. "B" - > "C" and "B" .xor. "C" - > "A" should give an original value of another operand.

2.3 Related research

The Paillier's cryptosystem was used in secure electronic voting [6]. The voter's ballot was encrypted, Paillier's encryption, by individual voter public key. This vote was signed, digital signature.Many voting officers do not know any information about ballots. At last, all of ballots were decrypted by voting director by using master private key.

An application to secure multimedia data in cloud environment using Paillier's cryptography [7], many multimedia contents were related to some sensitive event to owner, organization or even country endurance. Unfortunately, many contents need to be stored in untrustworthy cloud storage. Homomorphic encryption was used to conceal these contents in order to increase secrecy. These contents may be many times computed by many operators, legitimate or even not legitimate while they were kept invisible. These contents could be revealed by one who has a private key.

Nowadays, Kamal [8], many enterprises choose to perform their data processing on cloud computing. Data were encrypted with traditional encryption algorithms before sending to be processed in cloud server. Nevertheless, those encrypted data must be revealed before any data processing. Cloud computing officer could easily get plain data so that user's data were absolutely disclosed. Homomorphic encryption algorithm could prevent anyone from access plain data. All processing on cloud computing was still performed on normal data processing without data decryption.

Seddik [9] present method of protect text file from security breach. Data file was separated to many portions. Each data file portion was encrypted with defined encryption algorithm. Another data file perform encryption under different encryption algorithm. This technique increases confusion of cryptanalysis.

Hussain [10] suggested security protocol to manage security of customer data that got service of cloud storage provider. The designed protocol composes of key generation, key exchange and data encryption. Customer must encrypt their data file with their own invisible public key which was generated by key generation bot. No one else could reveal data file that store in cloud storage except one who have private key.

There are many problems about security such as costing and time delay in cloud storage security manipulation. Basma [11]presented protocol of encryption management based on Paillier's encryption algorithm. Automated master agent (AMA) was added to the multi agent system architecture (MASA) layer (cloud client-side). These agents were accountable for encryption and decryption performing in order to reduced time delay in data encryption. More ever, there will inform data owner if there were any occurring events about data manipulation on their own data by QR code under encoded key print.

3 Database Secure Manipulation

DSM-PHE protocol is conducted in overall explanation steps by steps as following.

3.1 Overall database secure manipulation scenario

Actors

In the normal situation, there are three actors involveddatabase manipulation. "Data owners (DOi:)", "Database administrator (DA:)" and "Database operator (DOP:)".

The scenario are brief details as following.

DA: is one who responsible in public, private key generation and encrypt, decrypt operation.

DOP: is one who undertook public key encryption (Paillier) and data manipulation.

DOi:'s data is firstly encrypted, by DA: with DOi:'s public key, and save it in database engine.

DOP: is a client who has manipulation responsibility about data processing without knowledge of its original plain data.

For some event, DOi:'s data is requested for declaration. DOi:s request, contain with plain data record ID, is sent to DA:. This requested encrypt data record is then retrieved from cloud storage or database engine.

DA: perform decryption on last cipher text, with DOi:'s private key (Paillier) so that plain data is revealed then this plain data is be further sent it back to DOi:.

There are two encryption operation responsibility undertaken by DA:.

First, DOi:'s data record is encrypted with DOi:'s public key (not Paillier). Second, DA: encrypt DOi:'s data record, with Paillier's encryption algorithm public key, before submit this data record to be kept in Database.

Iteratively, this data record is further performed data manipulation under designed DSM-PHE operation.

When DOi: make request to DA: for his own plain data then DA: would perform two tasks. First, decrypt cipher text, DOi:'s encrypted data record, with Paillier's encryption algorithm DOi:'s private key. Second, sent it to DOi:. After that, DOi: reveal plain text by decrypt it with his private key (not Paillier). The second task is compiled in order to preserve data secrecy from data transferring on network. This activity should use any public key system such as RSA.

Concisely, DSM-PHE operation as described later shall briefly explained in DSM-PHE therefore encryption tasks detail are mostly related to Paillier's encryption algorithm.

Scenario of DSM-PHE.

Detail of each step is explained in figure 1.

There were two database engines. First, DA:'s table look up is used to keep generating keys of all DOi:. Second, database engine is used to keep encrypted data records of all DOi:. All data record in database engine is always stored in encrypted from, all the time of data manipulation.

First, initial data is data record of data owner, DOi:, this data record is encrypted with DOi:'s public key (not Paillier), under DA:'s responsibility. After that, this data

is iteratively manipulated by database operator, DOP:, who is responsible in data operation such as addition, subtraction, multiplication and division. All data manipulation is processed on encrypted data record. DOP: has to encrypt data record, on encrypt data record, by DOi:'s public key (Paillier) which is retrieved from DA:'s table look up. However, DOP: can not access plain data all the time of his working.

Initial data is many times changed until data owner request for plain data reveal. Data owner, DOi:, inform DA: to perform this request. DA: must retrieve encrypted data record from database engine. DA: perform decryption on retrieved data record. DA: perform decryption on cipher, under Paillier's encryption algorithm, again with DOi:'s public key, not a DSM-PHE Paillier's encryption algorithm. DA: sent this plain text to data owner. DOi: perform decryption on sent cipher text with DOi:'s private's key (not Paillier) to gain a requested plain data record.

Details of all actor-tasks are presented in table 1. There are 14 main tasks or 23subtasks in total. Scenario of DSM-PHE, in table 1., is just simple protocols that proceeds only necessary task that serves on Homomorphic encryption.

For more secrecy, some task must be additionally provided such as authentication detection, plain data encryption while data transferring between DA: and DOi:. Customer certification, CA, is a possible technique to handle this task in order to obtain more authentications.

Since, there are many DOi: (i=1, n) thus DA: must create distinct public key and private key for each one. This couple of keys should be used in particular tasks as shown in table 1.

Normally, there will be many transactions that will proceed on a specific current data record. The data update operation may be addition, multiplication, division and subtraction. Therefore, DOP: must compile the right encryption algorithm with type of operation as explained above.

3.2 Key generation (practical example)

Data operation "addition and subtraction" that will explain in next topic are covered only Paillier's encryption exclude every task that are related to not Paillier's encryption.

DA: responsibility-key generation.

Choose p=3, q= 5; test required condition, gcd (3.5, (3-1).(5-1)) = gcd (15, 8)=1

Calculate for "n" value, n = p . q = 3 . 5 = 15

Calculate for λ value,$\lambda = lcm((3-1),(5-1)) = 4$

Calculate for μ value,$\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n$

Choose g= 4 then

$$\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n = (L(4^4 \bmod 1\,5^2))^{-1} \bmod n$$

$$a = L(4^4 \bmod 1\,5^2) = L(4^4 \bmod 1\,5^2) = \frac{4^4 \bmod 15^2 - 1}{15} = 2$$

Hence, $\mu = 2^{-1} \bmod 1\,5 = 8andgcd(a,n) = gcd(2,15) = 1.$

Therefore, public key is (n, g) = (15,4) and private key is$(\lambda, \mu) = (4, 2^{-1}\, mod\, 1\, 5 = 8)$.

Since, p and q are equivalent length therefore

$$n = p.q = 3.5 = 15$$

$$g = n + 1 = 15 + 1 = 16$$

$$\phi(n) = (p-1).(q-1) = (3-1).(5-1) = 8 = \lambda$$

$$\mu = \phi(n)^{-1}\, mod\, n = 8^{-1}\, mod\, 1\, 5 = 2$$

If "r" value is assigned as "4" then recheck condition of gcd (r,m). Since the result of gcd (4,15) =1 thus "r" =4 is useful. Therefore, "r" =4, public key is (n, g) = (15,16) and private key is $(\lambda, \mu) = (8,2)$.

3.3 Subtraction operation

Suppose that DOi:'s data is "3" and DOP: want to minus "3" with "2" or "3 - 2".

DA: responsibility.

DA:-4 starts with, DOi:'s data is encrypted once the first time and save this encrypted data in database engine.The "r" value is defined, suppose that r = 4.

Encrypt DOi:'s plain text ("3"), $c = g^m.r^n \bmod n^2 = 16^3.4^{15} \bmod 15^2 = 154$

DOP: responsibility.

DOP:-5 –retrieve DOi:'s encrypted data, $E(m1, r1)$, from database engine then hold this data for a moment.

DOP:-6 -request for Doi's: public key, (Paillier...), belong to a given specific data record ID owner, DOi:'s.

DOP:-7and 8 - Modular addition inverse of "2" (y) is prepared, call as "v".

$$y + v\, mod\, n = 0$$

$$v = n - 2\, mod\, n = 15 - 2\, mod\, 1\, 5 = 13$$

Perform encryption on "v", denoted as "m".

$$c = g^m.r^n\, mod\, n^2 = 16^{13}.4^{15}\, mod\, 1\, 5^2 = 79$$

Additional operation:

$$c = E(m1, r1).E(m2, r2)\, mod\, n^2$$

$$= 154.79\, mod\, 2\, 25 = 16$$

DA:-12 decrypt DOi:'s encrypted data with DOi:'s private key (Paillier...) toreveal plain text of "C" as following.

$$D(E(1m, 1r).E(2m, 2r)\, mod\, n^2) = L(c^\lambda\, mod\, n^2).\mu.mod\, n$$

$$= L(16^8 \, mod \, 2\,25)2. mod\, 1\,5$$

$$= \frac{121-1}{15}. 2 \, mod \, 1\,5$$

$$= 1$$

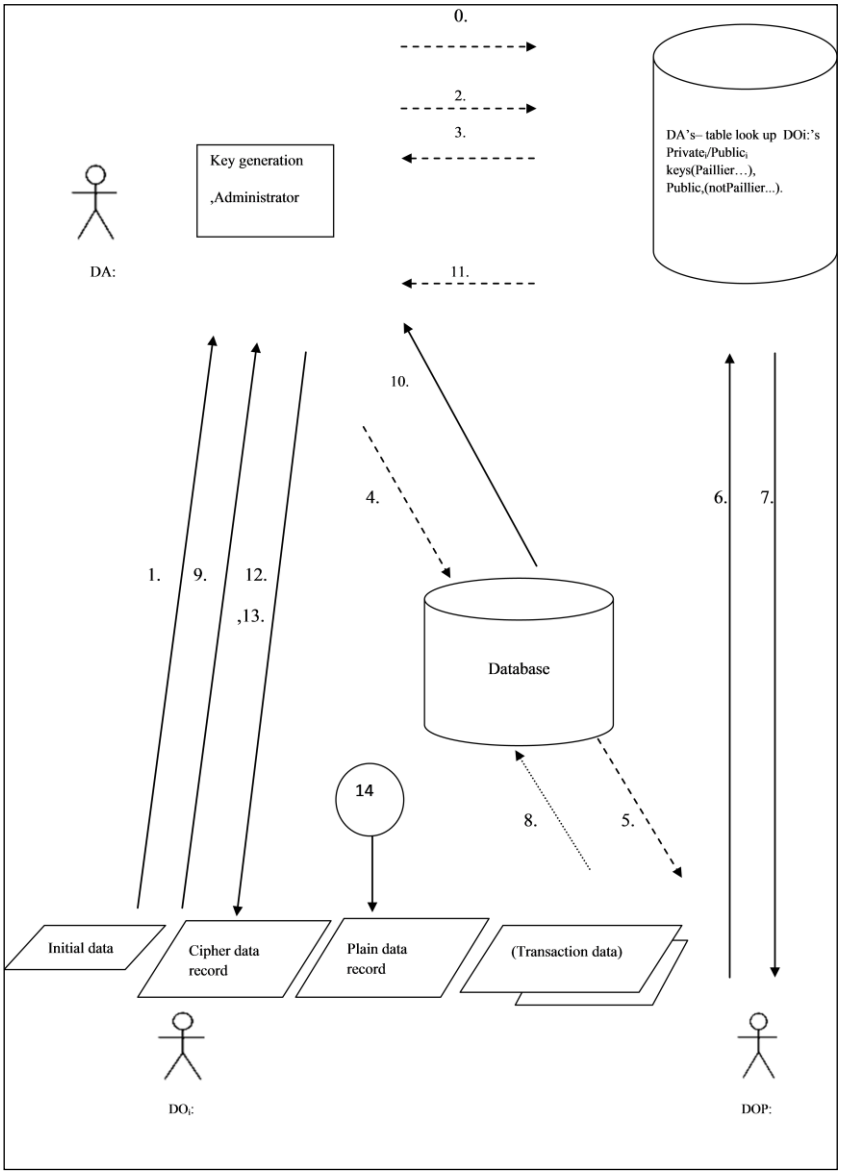

Fig. 1. Flow diagram of database secure manipulation based on Paillier's Homomorphic encryption (DSM-PHE)

Table 1. Actor's task of DSM-PHE

	Actor-task	Task description	Related actors or System
DA:	DA:-0	-perform Paillier's encryption algorithm public & private keys, for each DO_i:, keys preparation then kept in DA:'s table look up-Do_i's: public and private key (not Paillier..) were also generated. Only, public (not Paillier..) was sent to be kept while private key was sent to DO_i: in order to carefully look after.	DA:'s table look up
	DA:-2	-retrieve Do_i:'s public key (not Paillier..)	DA:'s table look up
	DA:-3	-get Do_i:'s public key -initial data record was encrypted by Do_i's public key (not Paillier..).	
	DA:-4	-encrypt it again by Do_i's public key (Paillier...) -concatenated it with plain primary key or plain data record ID. -send it to be kept in database engine.	Database engine
	DA:-10	-get data record ID from Do_i: sending request. -retrieve database engine for encrypted data record belong to a given specific record ID	Database engine
	DA:-11	-get DO_i:'s private key(Paillier...).	Database engine
	DA:-12	-decrypt Do_i:'s encrypted data record with Do_i:'s private key(Paillier...)	
	DA:-13	-send it back to DO_i: via networking.	DO_i:
DO:	DO_i:-1	-send plain initial data record, with plain data record ID, to DA:	DA:
	DO_i:-9	-sent request for data record disclosure, under specific data record ID, to DA:.	DA:
	DO_i:-14	-decrypt it with Do_i:'s private key (not Paillier..) to gain plain data record.	
DOP:	DOP:-5	Iteratively, perform transaction processing on encrypted data record under specific record ID. -retrieve database engine for encrypted data record belong to a given specific data record ID of request.	Database engine
	DOP:-6	-request for Do_i:'s public key, (Paillier...), belong to a given specific data record ID owner,Do_i:'s.	DA:'s table lookup
	DOP:-7	-got DO_i:'s public key,(Paillier...).	DA:'s table lookup
	DOP:-8	-define 'r2' value and check its validity. -calculate for additive or multiplicative inversion according to type of request operation. -encrypted data record was performed encryption belong to type of operation. -send this encrypted data record to be kept in database engine.	Database engines

3.4 Division operation

Suppose that DOi:'s data is "4" and DOP: want to divide it by "4" or "4/4".

DA: responsibility.

DA:-4 -Starting task, perform DOi:'s data encryption with DOi:'s public key (Paillier...) once the first time and save this data in Database. Choose "r" = 4; test gcd value,

$$\gcd(r,n) = \gcd(4,15)=1$$

Perform DOi:'s data encryption then save it in database engine.

$$c = g^m . r^n \, mod \, n^2$$

$$= 16^4 . 4^{15} \, mod \, 1\,5^2$$

$$= 214$$

DOP: responsibility.

DOP:-5 -retrieve DOi:'s encrypted data, $E(m1, r1)$, from database then hold this data for a moment.

DOP:-6 -request for Doi:s' public key, (Paillier...), belong to a given specific data record ID owner, DOi:'s.

DOP:-7& 8 -Modular multiplication inverse of "4" is prepared, call as "v".

$$y * v \, mod \, n = 1 \text{or}$$

$$v = y^{-1} \, mod \, n$$

$$= 4^{-1} \, mod \, 1\,5$$

$$= 4$$

Perform DOi:'s data encryption of divisor (4) with DOi:'s public key.

$$c = g^m . r^n \, mod \, n^2$$

$$= 16^4 . 4^{15} \, mod \, 1\,5^2$$

$$= 214$$

Perform multiplication ("4*(4-1)").

$$c = E(m2, r2)^{m1} \, mod \, n^2$$

$$= 214^4 \, mod \, 1\,5^2$$

$$= 16$$

DA:-12 responsibility.

DA: Decrypt DOi:'s encrypted data.

Reveal plain text of"C" by

$D(E(m1, r1)^{m2} \, mod \, n^2) = L(16^8 \, mod \, 1\,5^2).2 \, mod \, 1\,5 = 1.$

4 Performance Evaluation

4.1 Performance comparison

Assume that each task consumed one unit of time in computer operation. According to Database Secure Manipulation (DSM-PHE) explained above, there is greater number of tasks than ordinarydatabase manipulation, not any encryption activity.

Normal database manipulation, without any encryption, has not any overhead of key preparation so that its scenario of both subtractions and divisions perform just only five tasks.

DOi: - >DA: ->DOP: -> DA: - >DOi:

Scenario of both subtraction or division perform 14 main tasks or 23 sub tasks in total (in parentheses).

```
-Start-
//represent just one round operation excludes crypto
system
(not Paillier's encryption system).
DA:-0 (2 sub tasks)
- >DOi:-1(1) - >
- >DA:-2(1)- > DA:-3(2) - > DA:-4(3) - >
- >DOP:-5(1)- > DOP:-6(1)-> DOP:-7(1)-> DOP:-8(4) ->
- >DOi:-9(1) ->
- >DA:-10(2)-> DA:-11(1)-> DA:-12(1)-> DA:-13(1) ->
- >DOi:-14(1)
-End-
```

To summarize, DSM-PHE with total operations requires computing tasks about five times of normal database manipulation.

4.2 Key secrecy enhancement

Public & private, for each DOi:, keys that are prepared by DA: are kept in DA:'s table look up. These keys are managed and used by DA: and DOP:. If there is someone access to this file, then secrecy and integrity of DOi:'s data should be disclosed.

Since there are many operations on DSM-PHE, taking more processing time, thus additional processing task, in order to protect DOi:'s data security, must not spend too much processing time. To overcome this problem, DA: must undertake more tasks as explained in detail of additional task, as following explanation.

DA: define an arbitrarily random number (r) , has equivalent number of bit of target key, which is confidentially and no one can access it.

All public & private keys are processed through mathematical operation exclusive .OR. with an assigned random number (r) before the keys are kept.

DA: -prepare $^{(r)}$then transform each key, for example DOi:'s private key, equation 12, and public key, equation 13.

$$DO_i: key_{private} \oplus r \rightarrow key'_{private} \tag{12}$$

$$DO_i: key_{public} \oplus r \rightarrow key'_{public} \tag{13}$$

These transformed keys are kept in DA's table look up. In case of key request, DA: retrieve DOi:'s key from DA:'s table look up according to DOi:'s ID. This retrieved key must be transformed back to its original form. DA: just perform mathematical operation exclusive or, .XOR., with it.

$$DO_i: key'_{private} \oplus r \rightarrow key_{private} \tag{14}$$

$$DO_i: key'_{public} \oplus r \rightarrow key_{public} \tag{15}$$

As shown in equation 14 and 15, the transformed key willbe returned to original key. DA: could decide to create only one $^{(r)}$ value or r_i for each DOi: based on organization's security policy. Detail of all actor tasks are presented in table 1. There are 14 main tasks or 23 total tasks. In summary, there are five additional tasks in order to increase more data record secrecy. There are nine additional tasks for key secrecy enhancement, as shown in table 2.

Table 2. Actor's task of DSM-PHE (include DO_i's: key transformation-italic font)

		Actor-task	**Task description**	**Related actors or System**
Actor	DA:	DA:-0	*-assign a random number 'r'(same amount of bit as key size)* -perform Paillier's encryption algorithm public & private keys, for each DO_i:, keys preparation . *-transform keys to new transformed keys* -then kept them in DA:'s table look up - Do_i's: public and private key (not Paillier..) were also generated. Only, public (not Paillier..) was sent to be kept while private key was sent to DO_i: in order to carefully look after.	DA:'s table look up
		DA:-2	-retrieve Do_i's: public key (not Paillier..)	DA:'s table look up
		DA:-3	*-retrieve for Do_i's: public key (Paillier..)* *-transform Do_i's: transformed public key by exclusive .or. with assigned random number 'r'.* -get Do_i:'s public key (Paillier..) -initial data record was encrypted by Do_i:'s public key(not Paillier..).	DA:'s table look up
		DA:-4	-encrypt it again by Do_i's public key (Paillier...)	Database engine
			-concatenate it with plain primary key or plain data record ID. -sent it to be kept in database engine.	
		DA;-7	*-retrieve for Do_i's: public key (Paillier..)* *-transform Do_i's: transformed public key by exclusive .or. with assigned random number 'r'.* -send Do_i's: public key (Paillier..) to DOP:.	DA:'s table look up

		Actor-task	Task description	Related actors or System
		DA:-11	-get data record ID from Do_i: sending request. -retrieve database engine for encrypted data record belong to a given specific record ID	Database engine
		DA:-12	-get Do_i's: encrypted data record.	Database engine
			-retrieve for Do_i's: public key (Paillier..) *-transform Do_i's: transformed public key by exclusive .or. with assigned random number 'r'.*	DA:'s table look up
			-decrypt Do_i:'s: encrypted data record with Do_i:'s private key(Paillier...)	
		DA:-13	-send it back to DO_i: via networking.	DO_i:
	DO:	DO_i:-1	-send plain initial data record, with plain data record ID, to DA:	DA:
		DO_i:-10	-send request for data record disclosure, under specific data record ID, to DA:.	DA:
		DO_i:-14	-decrypt it with Do_i:'s: private key (not Paillier..) to gain plain data record.	
	DOP:	DOP:-5	Iteratively, perform transaction processing on encrypted data record under specific record ID. -retrieve database engine for encrypted data record belong to a given specific data record ID of request.	Database engine
		DOP:-6	*-request DA: for Do_i:'s: public key, (Paillier...), belong to a given specific* data record ID owner, Do_i:'s:.	DA:
		DOP:-8	-get Do_i's: public key, (Paillier...).	DA:'s table lookup
		DOP:-9	-define 'r2' value and check its validity. -calculate for additive or multiplicative inversion according to type of request operation. -encrypted data record was performed encryption belong to type of operation. -sent this encrypted data record to be kept in database engine.	Database engine

5 Summary and Suggestion

In summary, DSM-PHE consumed more computation tasks than normal database manipulation. Therefore, database administrator has to consider choosing DSM-PHE operation only in situation that sensitive data and secrecy are more important issues.

This paper presents computing on positive integer number while neglect positive, negative number of floating points number operation. Number theory and numerical method should be applied to solve variety kind of operands and decrease computational time problem.

Private and public key of each DOi:, which are stored in DA:'s– table look up, should be attacked by some intruder. Therefore, these series of keys could be enhanced their secrecy by simple hidden mathematical operation. These operations consume just only nine unit of computing time.

DSM-PHE model are composed of key generation and distribution module, encryptionanddecryption moduleanddatainversion module. These components work together to accomplish secure data manipulation on specific database. Even though

there are many tasks that all actors have to compiled but overall operation could reach data secrecy and integrity.

DSM-PHE should be a good solution of secure database manipulation especially when apply it on commercial cloud storage. It can prevent anonymous cloud database administrator deliberately access to customer database. Cloud database administrator can not disclose to kept data since they have no private key. This private key is not a static value since it is frequently changed by DA: by using key enhancement. DSM-PHE is suitable for numerical data processing. Hence, DSM-PHE is good for numerical data manipulation especial sensitive numerical data.

Nevertheless, DSM-PHE is also practical used in other disciplines such as national security, critical infrastructure control, health care, industry control system, inventory control and organization financial information. Personal behavior information on several social medias should be protected from social media provider since transactions are complete manipulated without reveal user's important data.

6 Acknowledgement

Database Secure Manipulation based on Paillier's Hormomorphic Encryption (DSM-PHE) has been supported from annual research funding, year 2015, of "Science and Technology Institute: (STRI) King Mongkut's University of Technology North Bangkok Thailand.

7 References

[1] C. Gentry, "A fully homomorphic encryption scheme," STANFORD UNIVERSITY, 2009.

[2] S. Choinyambuu, "Homomorphic Tallying with Paillier Cryptosystem," 2009.\

[3] S. Rao and T. David, Modular Arithmetic. CA: Standford University, 2009.

[4] P. S. Nordholt et al., "D2.1 State of the Art Analysis of MPC-Based Big Data Analytics," Eu, 2017.

[5] P. Cheung, "More Gates and their Applications," London, 2007.

[6] N. Pettersen, "Applications of Paillierrs Cryptosystem," Norwegian University of Science and Technology, 2016.

[7] N. S. Rao and S. R. Sree, "An Application to Secure Multimedia Data in Cloud Environment using Paillier Cryptography," Int. J. Comput. Trends Technol., vol. 17, no. 3, pp. 138–143, 2014 https://doi.org/10.14445/22312803/ijctt-v17p126.

[8] K. K. Chauhan, A. K. S. Sanger, and A. Verma, "Homomorphic Encryption for Data Security in Cloud Computing," in Proceedings - 2015 14th International Conference on Information Technology, ICIT 2015, 2016, pp. 206–209. https://doi.org/10.1109/icit.2015.39

[9] H. Seddik, "Combined Multi-encryption Techniques for Text Securing Using Block Cipher and Stream Cipher Crypto-systems," Int. J. Recent Contrib. from Eng. Sci. IT, vol. 1, no. 1, p. 53, Jul. 2013 https://doi.org/10.3991/ijes.v1i1.2901.

[10] M. E. Hussain and M. R. Hussain, "Securing Cloud Data using RSA Algorithm," Int. J. Recent Contrib. from Eng. Sci. IT, vol. 6, no. 4, p. 96, Dec. 2018 https://doi.org/10.3991/ijes.v6i4.9910.

[11] B. Hathout, S. Ghoniemy, and O. Ibrahim, "A modified cloud-based cryptographic agent for cloud data integrity," Int. J. Interact. Mob. Technol., vol. 11, no. 2, pp. 6–23, 2017 https://doi.org/10.3991/ijim.v11i2.6553.

8 Authors

Somchai Prakancharoen graduated doctoral degree in Information technology from King Mongkut's University of Technology North Bangkok, Thailand. His interestresearch topics are about computer security.

Article submitted 2019-07-29. Resubmitted 2019-09-22. Final acceptance 2019-09-23. Final version published as submitted by the authors.