EPiC Series in Computing
Volume 58, 2019, Pages 397–406
Proceedings of 34th International Conference on Computers and Their Applications

TrustedDW: A new Framework to Securely Hosting Data Warehouse in The Cloud

Kawthar Karkouda[1] , Ahlem Nabli[1] and Faiez Gargouri[1*]
[1] University of Sfax, Sfax, TUNISIA.
Kawthar.karkouda@gmail.com, ahlem.nabli@fss.usf.tn
faiez.gargouri@isims.usf.tn

Abstract

Nowadays cloud computing become the most popular technology in the area of IT industry. It provides computing power, storage, network and software as a service. While building, a data warehouse typically necessitates an important initial investment. With the cloud pay-as-you-go model, BI system can benefit from this new technology. But, as every new technology, cloud computing brings its own risks in term of security. Because some security issues are inherited from classical architectures, some traditional security solutions are used to protect outsourced data. Unfortunately, those solutions are not enough and cannot guarantee the privacy of sensitive data hosted in the Cloud. In particular, in the case of data warehouse, using traditional encryption solutions cannot be practical because those solutions induce a heavy overhead in terms of data storage and query performance. So, a suitable schema must be proposed in order to balance the security and the performance of data warehouse hosted in the cloud. In this paper, we propose (TrustedDW) a homomorphic encryption schema for securing and querying a data warehouse hosted in the cloud.

1 Introduction

Data warehouse is a specific database, which provides storage for huge amounts of historical data from heterogeneous operational sources. Those historical data will be used by analytical tools for producing business intelligence. For this purpose, data warehouse occupies a central place in enterprise business intelligence (BI). But building BI systems necessitate an important initial investment which may cause a problem while adopting this technology.

Cloud computing provides a new service which allows customers to create, maintain and query their data in the cloud while using their internet connection. This new service delivery model is lucrative for many companies. In fact, such services allow for outsourcing a DW and for running OLAP queries. Yet, data outsourcing brings out privacy concerns since sensitive data are stored, maintained and processed by an external third party that may not be fully trusted. A typical solution for preserving data privacy is by encrypting data locally before being sent in the cloud. Data security using cryptography is the most used solution. But this solution is not enough to secure data in the

[*] Created the first draft of this document

cloud because data will be decrypted before being executed by the providers. More than that, decrypting data in the cloud is a very expensive solution in terms of cost and time complexity. Those scenarios are not a suitable solution for data warehouse because of the high volume of data stocked in the warehouse and because of the nature of the OLAP query. Thus, processing the queries directly over encrypted data can significantly improve the query performance and the cost of using the cloud while maintaining data privacy. For this reason, we propose in this paper a new schema for sharing securely data warehouse in a multi- cloud. Our proposal is based on the homomorphic privacy presented in [1]. One serious deficiency of this homomorphic privacy is the possibility of being broken by clear text attacks. Thus, our contribution is to make this privacy homomorphism more robust and secure using a random splitting function, multi-cloud and perturbation value. It should be noted that it is not our aim to propose a solution as secure as the state-of-the-art encryption algorithms. We rather suggest a technique that provides a considerable level of overall security strength with respect to some performance overheads.

In this paper two signatures are proposed outer signature and inner signature for verifying the integrity of data sent to and received from the cloud.

This paper proceeds as follows: the second section surveys related works. The third section proposes a novel schema for securely hosting and querying a data warehouse in the cloud. The fourth section is devoted to shed light on some theoretical results. Finally, the paper ends with a conclusion.

2 Related works

Before outsourcing data to the cloud, most cloud providers propose to encrypt data with traditional encryption techniques. The symmetric and asymmetric encryption algorithms [2][3] are commonly used for such scenarios .The problem with those scenarios is that they are based on trust between the owner and the cloud provider, which is not the case because the cloud provider is not trust. Furthermore, the data and the keys are stocked in the same cloud provider. So, if intruders break the security system of the provider, they can steal the data with the keys and decrypt it easily. Moreover, the cloud provider must decrypt data before processing it. This operation is very costly mostly in the case of data warehouse.

Homomorphic encryptions seem to be more attractive for outsourced data to the cloud because it allows for running data in ciphertext in the cloud. RSA and ElGamal [4] cryptosystems are two multiplicative homomorphic schemas. Paillier [5] and Goldwasser –Micalli [6] are two additive homomorphic schemas. Authors in [7-8] propose a solution based on a fully homomorphic encryption. The idea is to find an encryption algorithm which can do addition and multiplication of encrypted data in ciphertext. The disadvantage of those algorithms is their high time complexity of encryption and decryption and the high volume of data generated after encryption.

Order preserving encryption [9] and multivalued Order preserving encryption MV-OPE [10] are used to perform computations over attributes that are used in the calculation of max and min aggregation functions or attributes that are compared using relational operators.

To maintain confidentiality and availability, some authors propose to store data at multiple cloud providers, such as DSky [11] and inercloud [12]. Those schemas use symmetric encryption tools to preserve confidentiality of data and distribute encrypted data over multiclouds.

Secret sharing schema [13] is very used in the cloud [14-15]. Most works use secret sharing to ensure the confidentiality and the disponibility of data by sharing data in multicloud. The problem with those schemas is the high volume overhead generated after encryption. That's why, authors in [16] propose a new model for sharing data warehouse inspired from secret sharing to remedy this problem. However, this approach suffers from high time complexity of decryption steps. Another problem when using this algorithm is that it cannot resist to collusion attack. To work out such a problem, authors in [17] propose S4 a new schema based on secret sharing for enforcing privacy in

Cloud data warehouse. The idea behind this proposal is to store secrets at one single CSP instead of sharing secrets to n CSP's. The privacy in S4 relies on the fact that k-1 splits are stocked in the CSP and the K^{th} splits necessary for reconstructing the secret is stocked in the owner. A playing this principal, they can avoid the problem of collusion, but the processing of the query cannot be done in the cloud.

In [18] the authors propose a new method for encrypting data warehouse hosted in the cloud. They propose to use several encryption techniques which allow the processing of analytical queries.

1.1 Discussion and motivation

The problem of data security is the most prevalent obstacle of adopting the cloud especially in the case of data warehouse. In fact, the latter stocks a high volume of data and need much time to process OLAP query. For this reason, in this paper, we propose a new schema that balances security and performance when outsourcing data warehouse in the cloud. Our schema is based on the simple privacy homomorphism described in [1]. It will be illustrated as it is given in [1]:

Let p and q be two large secret primes and $m= pq$ the product of such large secret primes. For that m is difficult to factor.

Consider the set of cleartext data $T = Zm$, and the set of cleartext operation $F= \{+_m ,-_m , \times_m \}$ consisting respectively of the addition, substraction and multiplication module m , with $m=pq$ .Let the ciphertext data set be $T' =Z_P \times Z_q$. Ciphertext operation F' is the component wise of these in F.

Define the encryption function $\phi(x) = [x \bmod p, x \bmod q]$. Given the two prime numbers p and q and the ciphertext $xp=x \bmod p$ and the ciphertext $x_q =x \bmod q$, the secret x is decrypted using the Chinese remainder theorem (CRT).

Our motivation behind the use of this privacy homomorphism is because it's based on the MOD operator. This is very interesting in terms of volume overhead. In term of confidentiality, the data will be encrypted with the two prime numbers p and q and will be computed in the range of $m = pq$. The decryption function is based on the use of Chinese remainder theorem. This technique is very practical in terms of temporal complexity.

The simple scenario is to encrypt data stocked in the data warehouse with the encryption function $\phi(x)$. After that, the cipher text data x_p and x_q will be sent to the cloud provider with the module m. The two prime numbers p and q will be kept secret in the owner. The data stocked in the cloud will be processed modulo m.

Unfortunately, this schema is not secure enough because the cloud provider can infer the two chunks of data and get the two secret parameters p and q. Malicious intruders can also break the security parameters of the cloud provider, get the encrypted data and the modulo m from the cloud provider and decrypt the data using the known cleartext attack as described in [19]. Consequently, a new way to sharing data warehouse will be suggested in the second section .

3 TrustedDW :A new framework to securely hosted data warehouse in the cloud

This section presents our new schema for hosting and querying data warehouse in the cloud securely. Our schema is composed of three parts to ensure the three levels of security CIA (confidentiality, integrity and availability). The first part, concerns our method to enforce the security of the algorithm used to ensure data confidentiality. The second part refers two new signatures, inner signature and outer signature, to verify the integrity of data sent and received from the cloud. The third part, present data sharing and data reconstruction process for TrustedDW. The global architecture of our new schema is described in figure 1.

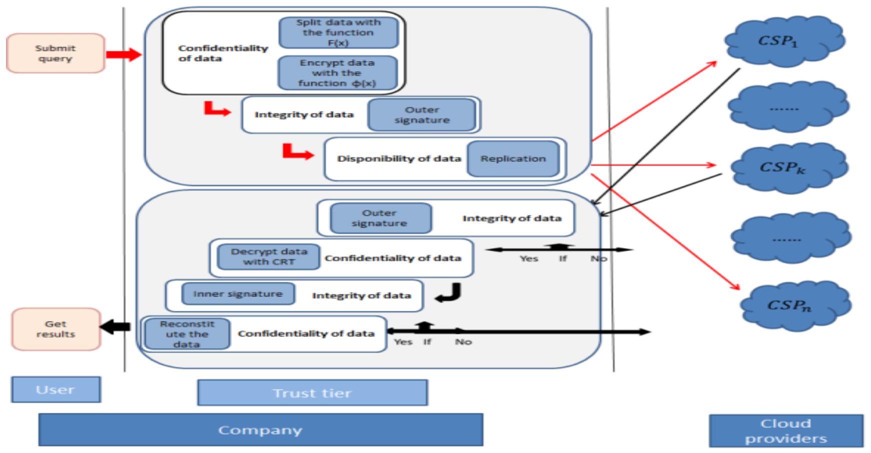

Figure 1Global architecture of TrustedDW

3.1 Enforcing the confidentiality of data

Firstly, to enforce the confidentiality of the homomorphic algorithm we propose to split the secret data X into a k small chunk with a random function F(x) such that:

$$F(x) = \sum_{i=1}^{k} x_i \qquad (1)$$

After that, we encrypt each chunk of data with the homomorphic function $\phi(x)$

Splitting data and encrypting each chunk of data separately with the homomorphic function $\phi(x)$ is not enough to secure sensitive data .To achieve this, we propose to stock each share of secret data in a different cloud provider. In this way, we can reduce the probability of inferring data and breaking the encryption function because each provider has only one chunk of the secret data. So, the problem of intern risk will be decreased. Likewise, it is difficult for malicious users to break the security parameter of k cloud providers at the same time and get the k chunks of secret data necessary to reconstruct the original data. So, the risk of breaking the system from an extern intruder will be diminished. This way, our model will be more secure in terms of confidentiality towards the cloud providers as well as from external attack.
Besides, when p, q, m = pq are very large integers, a small value x is very likely to have the same representation over Zm, Zp, and Zq that is x mod m = x mod p = x mod q if x< min (p, q). This is an undesirable feature because the homomorphic function $\phi(x)$ leaves the cleartext unencrypted (trivial ciphertext). To overcome this drawback, we propose to multiply secret data with two secret values rp and rq such that rp< p and rq<q. So our new homomorphic encryption function will be:

$$\begin{aligned}\phi(x) = (&[x_1 \times rp \bmod p,\ x_1 \times rq \bmod q],\\ &[x_2 \times rp \bmod p,\ x_2 \times rq \bmod q],\ \text{with}\ k \geq 2 \qquad (2)\\ &\ldots\ldots\ldots\\ &[x_k \times rp \bmod p,\ x_k \times rq \bmod q])\end{aligned}$$

After encrypting data with the homomorphic function $\phi(x)$, k pair of data will be produced. After that, each pair of chunk of data will be stocked in a cloud provider.

3.2 Ensuring the integrity of data

To ensure the integrity of data, we introduce in this paper new signatures named outer signature for verifying the integrity of share and inner signature for verifying the integrity of original data.

Outer signature

Outer signature is a new homomorphic function based on the modular approach.
For c = x mod n is equivalent to x-k = n*c so k is the rest of division .
So the signature of each share designed Sgn_x will be computed with the homomorphic function as :

$$Hs(x) = x \bmod n = Sgn\_x \qquad (3)$$

and we will compute the key of each signature as :

$$key\_x = x - (Sgn\_x \times n) \qquad (4)$$

After that, data will be sent to a CSP's.
When the owner receives data from the CSP's, he can verify his data with the two equations (3) and (4).

Inner signature

Inner signature is a self-checked signature based on the homomorphic propriety of modular approach. It's verified by computing the equivalence between the chunks of data generated after splitting the original data with the random function F(x) and the share of data received from the cloud. Algorithm 1 is used to verify the integrity of the original data in the owner after decrypting all share of data with the CRT and getting the original chunk of date x1, x2,… xk.

```
Algirithm 1 : Inner signature ($x_1$ ,$x_2$ ,$x_k$ ,$x_{1p}$ ,$x_{2p}$ ,$x_{kp}$ ,$x_{1q}$ ,$x_{2q}$ ,$x_{kq}$)
{ $S_p$=$|x_{1p} + x_{2p} +\ldots+ x_{kp}|_p$ , $S_q$=$|x_{1q} + x_{2q} +\ldots+ x_{kq}|_q$, $Chunk_p$=$|x_1 + x_2 +\ldots+ x_k|_p$ , $Chunk_q$=$|x_1 + x_2 +\ldots+ x_k|_q$
If (($S_p$ =$Chunk_p$) and ($S_q$=$Chunk_q$ )) Than
    Write ("Data is correct ")
 Else
    Write ("Data is not correct ")}
```

After computing the inner signature, if it's correct the trust tier reconstitutes the original data X with the function F(x), else it asks the cloud provider to get other share of data.

3.3 Scenario of sharing and reconstruction process of TrustedDW

Data sharing process :
- The trust tier person who is responsible for data security in the company proposes two secret prime numbers p and q and two secret values rp and rq, such that rp< p and rq< q . He also calculates the module m= pq and he chooses the parameter k which is the number of chunks of data generated after splitting the original data.

- He affects each IDk to a specific cloud provider. This is very important for maintaining the coherence of secret data.

- He splits the original data with the random function F(x) presented in equation (1).

- He encrypts the k chunks of data with the homomorphic function ϕ(x) presented in equation (2).

- The trust tier computes the signature of each chunk of data with the integrity function Hs presented in equation (3) and the keys of each signature with equation (4).

- Finally, he replicates each chunk of data with its signature and its key and sends them to the corresponding cloud provider.

The scenario of data sharing process is presented in figure 2:

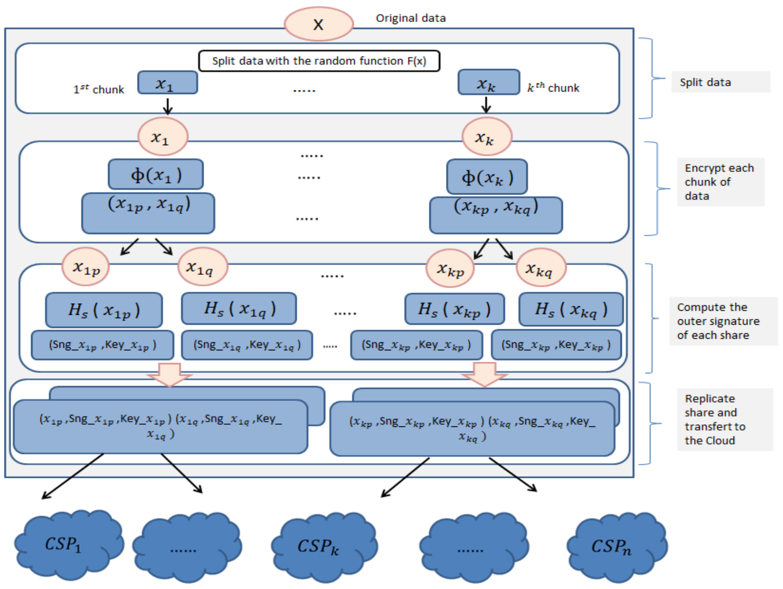
X
Original data
Split data with the random function F(x)
1^{st} chunk
x_1
.....
x_k
k^{th} chunk
Split data
x_1
x_k
$\phi(x_1)$
(x_{1p}, x_{1q})
$\phi(x_k)$
(x_{kp}, x_{kq})
Encrypt each chunk of data
x_{1p}
x_{1q}
x_{kp}
x_{kq}
$H_s(x_{1p})$
$H_s(x_{1q})$
$H_s(x_{kp})$
$H_s(x_{kq})$
(Sng_x_{1p} ,Key_x_{1p})
(Sng_x_{1q} ,Key_x_{1q})
(Sng_x_{kp} ,Key_x_{kp})
(Sng_x_{kp} ,Key_x_{kp})
Compute the outer signature of each share
(x_{1p} ,Sng_x_{1p} ,Key_x_{1p}) (x_{1q} ,Sng_x_{1q} ,Key_x_{1q})
(x_{kp} ,Sng_x_{kp} ,Key_x_{kp}) (x_{kq} ,Sng_x_{kq} ,Key_x_{kq})
Replicate share and transfert to the Cloud
CSP_1
......
CSP_k
......
CSP_n

Figure 2 Data sharing process

Data reconstruction process :

-The trust tiers ask k cloud providers to get the shares of corresponding data X (x_{1p}, Sgn_x_{1p}, key_x_{1p}) and (x_{1q}, Sgn_x_{1q}, key_x_{1q}) from CSP1, (x_{2p}, Sgn_x_{2p}, key_x_{2p}), (x_{2q}, Sgn_x_{2q}, key_x_{2q}) from CSP2, (x_{kp}, Sgn_x_{kp}, key_x_{kp}) and (x_{kq}, Sgn_x_{kq}, key_x_{kq}) from CSPk .

- He verifies the correctness of each share with the signatures and the keys. In case of errors, the trust tier can ask CSP's to get a new share.

-He computes the scalar product of each share (x_{1p}, x_{1q}) , (x_{2p}, x_{2q}) and (x_{kp}, x_{kq}) by (r_p^{-1} *mod p*, r_q^{-1} *mod q*) to retrieve (x_1 *mod p*, x_1 *mod q*),, (x_2 *mod p*, x_2 *mod q*) and (x_k *mod p*, x_k *mod q*) .

-After that, the trust tier decrypts the data using the Chinese remainder theorem with the two secrets parameters p and q and with the shares of data (x_1 *mod p*, x_1 *mod q*), , (x_2 *mod p*, x_2 *mod q*) and (x_k *mod p*, x_k *mod q*) .

- He verifies the integrity of the original data with the inner signature.

-If the inner signature is correct, the trust tier computes the original data with the function F^{-1}(x). If not, he asks the cloud provider to get another share of data.

The scenario of data reconstruction process is presented in figure 3:

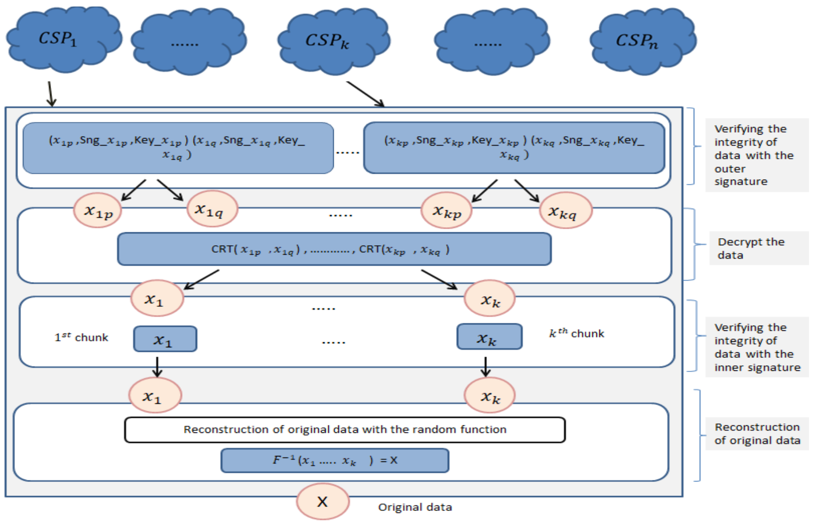
CSP_1
......
CSP_k
......
CSP_n
(x_{1p} ,Sng_x_{1p} ,Key_x_{1p}) (x_{1q} ,Sng_x_{1q} ,Key_ x_{1q})
.....
(x_{kp} ,Sng_x_{kp} ,Key_x_{kp}) (x_{kq} ,Sng_x_{kq} ,Key_ x_{kq})
Verifying the integrity of data with the outer signature
x_{1p}
x_{1q}
x_{kp}
x_{kq}
CRT(x_{1p} , x_{1q}) , ..., CRT(x_{kp} , x_{kq})
Decrypt the data
x_1
x_k
1st chunk
kth chunk
Verifying the integrity of data with the inner signature
Reconstruction of original data with the random function
F^{-1}(x_1 ..... x_k) = X
Reconstruction of original data
X
Original data

Figure 3 Data reconstruction process

4 Security analysis and performance evaluation of our schema

This section is devoted to illustrate the relevance of our approach along two axes. The first axis is about the security features of our scheme and the second axis is about the performance of our schema in terms of time complexity and volume overhead when using the pay-as-you go paradigm.

4.1 Security analysis

Confidentiality of data

To protect data from plaintext attacks and from malicious cloud providers, we propose to split data into k chunks with the random function F(x) and encrypt each chunk with the homomorphic function $\phi(x)$presented in equation 2. Then, each encrypted chunk will be stocked in a different cloud provider.

The objective is to keep minimal information about data and parameters among the cloud provider. As a result, we can reduce the risk of inferring the data and breaking the system.

The role of trust tier as a middle tier between the user and the cloud is an effective solution which guarantees the confidentiality of the two secrets p and q.

Proof:

If the cloud provider predicts the p and the q or gets the p and the q (worst case), he can predict x as

$X= y \times p + x_p$ and $X= y \times q + x_q$ such that $x<m$;

We can conclude that even if the two secret parameters are discovered by the cloud provider, he cannot identify whether the chunks of data correspond to the parameters p or *q or not*. This ambiguity can disrupt the work of inferring the data. Furthermore, since factoring is hard, inferring the two shares of data without known whether the chunks of data correspond to the parameters p or q cannot be done in the case of a huge volume of data as in the data warehouse.

In the worst case, if the cloud provider infers the data, decrypts the entire share and gets original chunk, the security of our original data cannot be breached because the cloud provider has just a chunk of data and not all the original data. So, we can conclude that splitting data is essential to the security of our schema. That's why the parameter k will be chosen carefully and should be $k \geq 2$.

Similarly, it can be argued that the confidentiality of our schema is better with this new sharing strategy. In fact, even if the malicious intruder gets the two secret parameters p and q, it is hard to break the security of k cloud providers and get the chunks of secrets at the same time. Even if he does this and decrypts the k shares of data, he cannot reconstruct the original data because he doesn't know the random function F(x).

Integrity of data

Outer signature

In our schema the processing of data is done in ciphertext. So there is no need to use a complex integrity function for ensuring the security of data. Our goal is to propose a simple method allows the verifying of the data sent and received from the cloud with minimum time complexity. Our integrity function $H_s(x)$ is homomorphic. This is very practical in the case of data warehouse.

Inner signature

The inner signature is a self-checked signature. It is based on the homomorphic characteristic of arithmetic modular. So, if there are some mistakes that cannot be detected with the outer signature, the inner signature can detect it.

Collusion

The main advantage of our work is that the risk of collusion is small because data is split randomly with the random function F(x). Thus, it is hard for the k clouds providers to predict how data is split if it colludes.

4.2 Performance evaluation

Volume overhead

Our encryption function is based on the MOD operator which divides the data in a small residue number. So, there is no overhead volume when encrypting initial data. The original data is split into k chunks and each chunk will be encrypted with the Mod operator two times. The volume of the encrypted data cannot exceed twice the volume of the original data.

Temporal complexity

In our scheme for the encryption phase, we need just O(n) operation because the encryption function needs only modular operation. The decryption function is based on the CRT. So we just need O (lglg n) operation for the decrypting phase.

Comparison of our schema with the existing related approaches

In this section, we compare our schema with the approaches presented in our state of the art in terms of security and performance. Tables 1 synthesize the features of some approaches discussed above.

		[15]	[18]	[16]	Our schema
Confidentiality		yes	yes	yes	yes
Integrity	Outer signature	No	No	yes	yes
	Inner signature	No	No	yes	yes
Disponibility		yes	No	yes	yes
Collusion		yes	No	yes	No
OLAP query		yes	yes	yes	yes
Range query		No	yes	No	No
Aggregation function		yes	yes	yes	yes
Data volume		6n	-	3n	6n
Time complexity of encryption phase		O(n)	-	O(n)	O(n)
Time complexity of decryption phase		$O(nlg^2n)$	-	$O(nt^2)$	O(lglg n)

Table 1 Comparaison of Data warehouse sharing approaches

5 Conclusion and future work

In this paper, we introduced TrustedDW as a new framework for sharing data warehouse in multiclouds. This schema ensures the security of data warehouse hosted in the cloud with reasonable time complexity and minimum volume overhead. Our proposed schema is based on a homomorphic encryption algorithm that reveals a serious weakness as it can be deciphered by ciphertext attacks. To overcome this weakness, we propose a new method of using this homomorphic privacy based on splitting data, multi cloud providers and perturbation values. With this new sharing schema, we can reduce the risk of breaking the security of the system by both cloud providers and malicious intruders. To ensure the integrity of data, two new signatures are proposed, outer signature and inner signature. As future work, we plan to propose an index to support processing of the range query in the cloud providers because this operation is done by the owner after decrypting all the data in the case of our schema. We eventually endeavour to evaluate our schema in real cloud providers

References

[1] Rivest R L, Dlemam L A , .Dertouzos M D (1978) on data bank and privacy homomorphisms .Foundations of secure computation pp 169-177.

[2] Popa R A, Redfield C M S, Zeldovich N, Balakrishnan H (2012) CryptDB :Processing queries on an encrypted database .ACM 103-111 .

[3] Liu D (2014) Securing outsourced databases in the cloud" .In security, privacy and trust in cloud systems . Springer,Heidelberg age 259-282 .

[4] Gamal T E (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. In Proceedings of CRYPTO 84 on Advances in cryptology, pages 10–18. Springer-Verlag New York, Inc.

[5] Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. In 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic, volume 1592.

[6] Bringe J, Chabanne H, Izabachene M , Pointcheval D, Tang Q,Zimmer S (2007) An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication. Springer-Verlag

[7] Jyh-Haw Y (2016) A Secure Homomorphic Encryption Algorithm over Integers for Data Privacy Protection in Clouds .SmartCom 2016

[8] Zhao F, Li C , Chun F L (2014) A cloud computing security solution based on fully homomorphic encryption . 16th international conference on advanced communication technology IEEE

[9] Agrawal R , Kiernan J, Srikant , Xu Y (2004) Order preserving encryption for numeric data .Sigmod june 13-18 paris france

[10] Kathen H, Amagasa T, Kitagawa H (2010) MV-OPES : Multivalued-order preserving encryption schemes :A novel scheme for encrypting integer value to many different values . IEIC Trans.

[11] Bessami A, Correia M, Quaresma B, André F, Sousa P (2011) DepSky :dependable and secure storage in the cloud –of-clouds .In processings of the sixth conference on coputersystems.ACM,31-46.

[12] Caclin C, Haas R, Vukolic M (2010) Dependable storage in the intercloud .IBM rechearch , vol.3783,1-6

[13] Adi S (1979) how to share a secret. Communication of the ACM novembre ,volume 22.

[14] Mohammad A, Ernesto D, Rasool J, Stelvio C, Zeinab G (2012) AS5: A Secure Searchable Secret Sharing Scheme for Privacy Preserving Database Outsourcing. 7th International Workshop, DPM

[15] Karkouda k, Harbi N, Darmont J, Gavin G (2012) Confidentialité et disponibilité des données entreposées dans les nuages. 9ème atelier Fouille de données complexes (EGC-FDC) Bordeaux, France.

[16] Varunya A, Harbi N, Darmont J (2015) A novel multi secret sharing approach for secure data warehousing and On-Line analysis processing in the cloud. IGI

[17] Moghadam S S, Darmont J, Gavin G (2017) S4: A New Secure Scheme for Enforcing Privacy in Cloud Data Warehouses. (ICIST 2017), Mar 2017, Dubai, United Arab Emirates. pp.9-16

[18] Claudivan C L, Valéria C T, Matwin S, Rodrigues R C, Aguiar C D C (2014) Processing OLAP Queries over an Encrypted Data Warehouse Stored in the Cloud ..DaWaK 2014, LNCS 8646, pp. 195–207, 2014

[19] Ervest F B, Yacov Y (1987) on privacy homomorphism's .advances in cryptology eurocrypt,