Towards Cloud Data Warehouses of Multivalued Encrypted Values

Claudivan Cruz Lopes[1], Valéria Cesário Times[2], Stan Matwin[3]

[1] Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Brazil
claudivan@ifpb.edu.br
[2] Universidade Federal de Pernambuco, Brazil
vct@cin.ufpe.br
[3] Dalhousie University, Canada
stan@cs.dal.ca

Abstract. The motivation behind generating multivalued encrypted values by multivalued encrypted techniques (METs) is the minimization of encrypted data redundancy. In the literature, there are METs that enable the processing of sum aggregations, selection constraints and sorting operations over multivalued encrypted values. However, we found a lack of METs for processing data groupings over multivalued encrypted values. In this article, we propose two novel METs that allow the execution of data groupings over multivalued encrypted values and we specify two novel data schemas for encrypted DWs (MV-HOM and MVSE-HOM) that allow analytical queries with data groupings be performed over multivalued encrypted values. Also, we conducted a performance evaluation of encrypted DWs stored in a cloud and results showed that MV-HOM's overhead was decreased up to 10.41% (with 16 nodes) when compared to a non-encrypted dimensional data schema, while the overhead imposed by MVSE-HOM was high (89.91% with 16 nodes).

Categories and Subject Descriptors: H.2.7 [**Database Management**]: Database Administration—*Security, integrity and protection, Data warehouse and repository*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation*; E.3 [**Data**]: Data Encryption

Keywords: Data Warehouse, Analytical Queries, Multivalued Encrypted Values

1. INTRODUCTION

Traditional security mechanisms of currently database management systems (DBMS), which are mainly based on authentication and authorization, have become insufficient to protect sensitive data, especially if the database is hosted in an untrusted server, such as a *Database as a Service (DaaS)* provider [Suciu 2012]. In this case, data confidentiality may be not achieved when data are stored in their original format, which can be read and interpreted. A solution to improve data confidentiality is to encrypt sensitive data in a trusted environment before sending them to be stored in the unsafe server. Thus, even if an adversary obtains access to the data, he will be unable to interpret them because they are encrypted [Vimercati et al. 2010].

Several studies propose the use of encryption techniques to encipher sensitive data, which allow operations (comparisons and calculations) be performed over encrypted data, enabling the execution of queries over them without requiring decryption [Popa et al. 2012; Chen et al. 2011; Shi et al. 2007; Hore et al. 2012; Tu et al. 2013]. Among these techniques are those that encipher unencrypted equal values into distinct encrypted values with high a probability [Liu 2014; Kadhem et al. 2013]. We

This work is supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) under the grants 246688/2012-2 and 246263/2012-1, by the Natural Sciences and Engineering Research Council of Canada, and by the Canadian Bureau for International Education

designate these techniques as *multivalued encryption techniques* (METs), and the encrypted values generated by METs are called *multivalued encrypted values*.

The motivation behind generating multivalued encrypted values by METs is the minimization of encrypted data redundancy, and consequently, the enhancement in protection against *statistical attacks* [Kadhem et al. 2010]. A statistical attack occurs when an adversary uses some statistical function, such as the repetition frequency of encrypted data, in order to try to infer the unencrypted values. The protection against statistical attacks is an important issue for the encryption of high-redundant databases, such as data warehouses (DWs). This is essential because if the DW's data are encrypted based on fixed encrypted values, then a high degree of encrypted data redundancy is found in this DW, implying in a vulnerability that can be exploited by statistical attacks. Thus, the use of METs in the encryption of a DW helps to minimize the redundancy of the DW's encrypted data.

Currently, there are some METs that enable the processing of sum aggregation functions [Boneh et al. 2013; Ge and Zdonik 2007] and the execution of selection constraints and sorting operations [Kadhem et al. 2010; Liu and Wang 2012] over multivalued encrypted values. However, we found a lack of METs for allowing the processing of data groupings over multivalued encrypted values. A data grouping means that equal values belonging to a same attribute, and retrieved by an analytical query, are grouped into a single value (i.e. eliminating their recurrence). Since multivalued encrypted values produced by current METs are different from each other with a high probability, then they do not enable the creation of data groupings because groups of multivalued encrypted values are probable groups of individual values. Therefore, the proposal of data encryption schemas for DW by using METs and the performance evaluation of analytical queries that include data groupings over these schemas constitute the focus of this article.

Contributions of our work are enumerated as follows.

(1) We propose two novel METs that enable the execution of data groupings over multivalued encrypted values.
(2) Based on our MET proposals, we specify two novel dimensional data schemas for encrypted DWs, called MV-HOM and MVSE-HOM, which allow analytical queries based on data groupings be performed over multivalued encrypted values.
(3) By using the Star Schema Benchmark (SSB) [O'Neil et al. 2009], we conducted a performance evaluation of encrypted DWs stored in a cloud and modeled according to MV-HOM and MVSE-HOM, in terms of analytical query processing. Results showed that MV-HOM produced performance gains w.r.t. MVSE-HOM in the execution of data groupings over multivalued encrypted values. Also, MVSE-HOM presented a high overhead when compared to a non-encrypted DW, while the greater was the parallelization of the processing of analytical queries, smaller was the overhead imposed by MV-HOM w.r.t. a non-encrypted dimensional data schema.

This article is organized as follows. Section 2 discusses related work. Section 3 explains the problem to be solved. Section 4 describes the novel MET proposals that enable data groupings over multivalued encrypted values. Section 5 details the novel encrypted dimensional data schemas MV-HOM and MVSE-HOM, while Section 6 reports the performance results of analytical queries processing over the proposed data schemas. Finally, Section 7 concludes the article and addresses future work.

2. RELATED WORK

The literature reports the use of several METs that allow operations be performed over multivalued encrypted values. *Homomorphic Encryption* (HOM) [Boneh et al. 2013] allows some operator *op* be computed over multivalued encrypted values. Thus, when $ENC(x)\ op\ ENC(y) = ENC(x\ op\ y)$ and $DEC(ENC(x)\ op\ ENC(y)) = x\ op\ y$, then *ENC* and *DEC* are considered as homomorphic encryption and decryption functions, respectively [Castelluccia et al. 2009]. The HOM encryption has been used

in the processing of aggregation functions, such as *sum* and *avg* [Ge and Zdonik 2007], and calculation of statistical functions, such as *standard deviation* and *logistical regression* [Lauter et al. 2011], over multivalued encrypted values. Also, we found a proposal of a *fully homomorphic encryption* (FHE) [Gentry 2010], which is able to compute any arbitrary operator (such as addition, multiplication, AND, OR and XOR), and hence, any function, over multivalued encrypted data. However, researchers have reported that FHE encryption have impractical performance, preventing its applicability in real applications [Wang et al. 2012; Gahi et al. 2011].

In the *Multivalued Order Preserving Encryption* (MV-OPE) [Kadhem et al. 2010], the encryption is performed so that the generated multivalued encrypted values preserve the same order as their corresponding unencrypted values. Thus, given an MV-OPE encryption function *ENC*, when $x < y$, then *ENC(x)* $<$ *ENC(y)* and *ENC(x)* $\neq$ *ENC(y)*. The MV-OPE encryption has been used to enable the computation of the following operations over multivalued encrypted values: sorting operations, comparisons using relational operators, such as $=$, $>$, $<$, $\geq$, $\leq$ and $\neq$, and the aggregation functions *max* and *min* [Liu 2014; Kadhem et al. 2013; Liu and Wang 2012]. However, current MV-OPE proposals do not enable the processing of data groupings over multivalued encrypted values.

In the *Searchable Encryption* (SE) [Shi et al. 2007; Boneh et al. 2004; De Caro et al. 2010], an encryption function *ENC* is performed over pairs *(point, record)*, so that *ENC(point, record)* $=$ *m*, where: *point* corresponds to the values of some attributes of a data record, elected to be the *searchable attributes*; *record* is a data record itself; and *m* is a resulting multivalued encrypted value. SE encryption enables the execution of conjunctive, subset and range conditions over multivalued encrypted values [Boneh and Waters 2007]. For this, SE encryption defines a function *QUERY* for searching multivalued encrypted values that fall in a given *n-dimensional* space. That is, *QUERY(m, S)* $=$ *record*, where: *m* is a multivalued encrypted value to be queried; *S* is a given n-dimensional space, and *record* is a resulting data record whose *point* belongs to *S*. However, the aforementioned conditions is performed over searchable attributes altogether, impeding the execution of conditions specified over some searchable attributes or even over attributes that were not elected as searchable attributes. Also, SE encryption is unable to process data groupings over multivalued encrypted data.

3. PROBLEM DEFINITION

We are interested in solving the following problem: *how analytical queries can be executed over a DW of multivalued encrypted values?* The motivation for addressing this problem is illustrated as follows.

Consider a DW application concerned to sales orders whose star schema is depicted in Figure 1, where *Date*, *Customer* and *Product* are dimension tables, and *Order* is a fact table. In the dimension tables, the attributes with suffix *Key* are primary keys, while the remaining ones are descriptive attributes. In the fact table, *OrderKey* is the primary key, and *OrderDate*, *OrderCustomer* and *OrderProduct* are foreign keys to the dimension tables *Date*, *Customer* and *Product*, respectively;

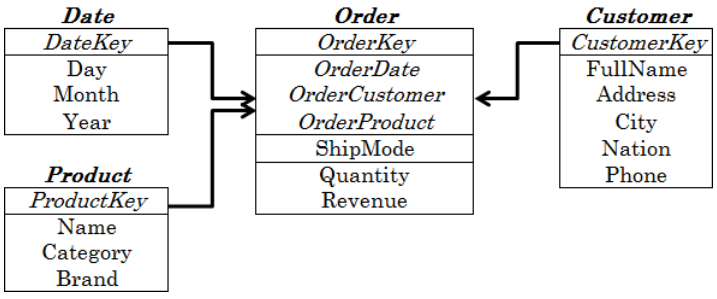

Fig. 1. An example of a star schema

while *ShipMode* is a descriptive attribute of a fact, and *Quantity* and *Revenue* are measures. Also, a fragment of a corresponding DW encrypted by METs is shown in Figure 2. We consider that in this encrypted DW, all attribute values are multivalued encrypted values.

Consider an analytical query that lists the sum of the sales orders' revenue grouped and sorted by product category and by month. Selection constraints are used to filter sales orders made between *January 2014* and *March 2014*, and whose number of items sold is greater than 25 units. When this query is executed over the encrypted DW of Figure 2, the sum of revenue values, the filter constraints applied over the values of the attributes month and quantity, the sorting and grouping of category values and of month values, and joins between the fact table *Order* and the dimension tables *Date* and *Product* should be performed directly over the multivalued encrypted values stored in the DW. However, the application of existing METs for encrypting a DW does not enable the DBMS's query processor to create groups of encrypted data. This occurs because all encrypted values are distinct from each other, and the current METs do not define how data groupings are processed by considering the indistinguishability of encrypted values.

Thus, our work focuses on the definition of METs for processing data groupings over multivalued encrypted values, and on the design of encrypted dimensional data schemas that enable DBMS's query processors to perform analytical queries over a DW encrypted by METs (i.e. the processing of aggregation, selection, join, sorting and data grouping).

4. NOVEL MET PROPOSALS FOR SUPPORTING DATA GROUPINGS

Our MET proposals allow the processing of data groupings over multivalued encrypted values by re-encrypting them at query runtime, so as to transform multivalued encrypted values into new encrypted values that can be grouped by a query processor. As the re-encryption is done at runtime, only the encrypted values selected by a query processor and stored in main memory are re-encrypted, whereas those that were not selected remain multivalued in the encrypted database. Regarding data confidentiality, the execution of queries is still performed over encrypted data and does not reveal unencrypted values because the multivalued encrypted values are not decrypted during query processing.

We define two novel METs based on existing encryption techniques: the MV-OPE encryption was used as a basis for our first MET definition that is detailed in Section 4.1, while the SE encryption was used as a basis for specifying our second MET proposal defined in Section 4.2.

Date

Remaining attributes	Month
...	a1!281
...	8b7#18
...	29g$18
...	78%1d2
...	6&5d71
...	7*e012

Product

Remaining attributes	Category
...	3q4*51
...	98w72&
...	r768%5
...	98g$12
...	5f#281
...	76#2j8
...	6@5h94
...	2!221k

Order

Remaining attributes	Quantity	Revenue
...	8h!176	41c@02
...	69@h01	55$8m8
...	44f3#3	#41b33
...	2s$187	3@1v44
...	101f%0	8z!934
...	31n87&	1a&234
...	61c2!3	9s67*!8
...	56z#38	67d8%9
...	375j*6	543h0#
...	8g$790	98$7j6
...	3@n789	498?t7
...	62t%34	856r@7
...	77?m89	9?78y1
...	13f&21	2&0y98
...	89s*01	8$06h0
...	1?23a1	k90?02

Fig. 2. A fragment of a DW of multivalued encrypted values

4.1 A Novel MV-OPE-based MET

Our first MET proposal defines a function ENCRYPT that generates a *list of ordered integer values* for an unencrypted value, so as to obtain a multivalued encrypted value chosen randomly from this this list. For the sake of simplicity, we adopt the notation *list* to refer to a list of ordered integer values. Further, we assume that the *list size* is defined properly so that a good randomness is achieved for selecting an integer value (i.e. a multivalued encrypted value) from this list.

A list is delimited by $[S \dots L]$, where S and L are computed by ENCRYPT, and represent the *smallest* and the *largest* values of the list, respectively. Figure 3 illustrates these lists for the values of the attribute *Month*, which depicts months of years. For example, the application of ENCRYPT over the value '2014/01' results in a multivalued encrypted value that is picked randomly from the list $[10 \dots 40]$. Inversely, our first MET defines a function DECRYPT that transforms any multivalued encrypted value from a list into the respective unencrypted value. For instance, in Figure 3, when DECRYPT is applied over any value from the list $[10 \dots 40]$, the value '2014/01' is calculated and returned by DECRYPT.

Moreover, each list is *labeled*. Thus, there is an association between unencrypted values, lists and labels, such that: each unencrypted value is associated with a list; each list is associated with a label; and each label is associated with an unencrypted value. For example, in Figure 3, the value '2014/01' is associated with the list $[10 \dots 40]$ whose label is *Label1*; and the *Label1* is associated with the unencrypted value '2014/01'.

For multivalued encrypted values preserving the same order as their respective unencrypted values, our first MET guarantees two properties: *(i)* given any unencrypted values t_i and t_j, if t_i precedes t_j, then L_i precedes S_j, where L_i is the largest value of the list for t_i, and S_j is the smallest value of the list for t_j; and *(ii)* given any unencrypted values t_i and t_j, if $t_i < t_j$, then $La_i < La_j$, where La_i and La_j are the labels associated with t_i and t_j, respectively.

Multivalued encrypted values produced by our first MET allow the computation of selection constraints, data groupings and sorting operations directly over these values by the DBMS's query processor. The values S and L are used in selection constraints. For example, a selection constraint *Month between '2014/01' and '2014/03'* specified in a query is mapped to *EncMonth between 10 and 120*, where *EncMonth* is equivalent to the attribute *Month*, so that its values are encrypted by our first MET; and the values 10 and 120 are the smallest and the largest values of the lists defined for the values '2014/01' and '2014/03', respectively, as shown in Figure 3.

Regarding the labels used in data groupings and sorting operations, our first MET defines a function GROUP that re-encrypts multivalued encrypted values by transforming them into labels that are associated with their respective unencrypted values. For example, in Figure 3, any multivalued encrypted value generated from the list $[10 \dots 40]$ is transformed by GROUP into *Label1* (which is associated with the value '2014/01'). As *Label1* is the same for each multivalued encrypted value chosen from the list $[10 \dots 40]$, this allows the computation of data groupings over labels (i.e. encrypted values) by the query processor. Also, because labels preserve the same order as their corresponding unencrypted values, the groups of labels can be sorted by the query processor.

The function GROUP must be deployed as a *stored procedure* in an encrypted database, and must be invoked at query runtime. When a query is issued by a user, it must be mapped into another

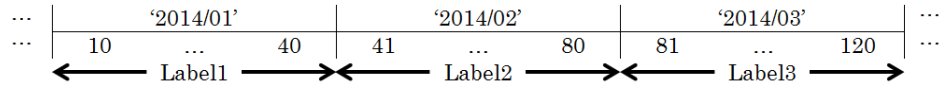

Fig. 3. An example of lists

query that calls GROUP. For example, the *projection*, *group by* and *order by* clauses specified in a SQL query as *select Month ... group by Month order by Month* are mapped to *select GR(EncMonth) as Mon ... group by Mon order by Mon*, where: *GR* is a name given to the stored procedure that implements GROUP; *EncMonth* is equivalent to the attribute *Month*, so that its values are encrypted by our first MET; and *Mon* is an alias for *GR(EncMonth)*. Thus, at runtime, the query processor computes the stored procedure *GR* for each multivalued encrypted value selected by the query, in order to create sorted groups of labels.

The labels resulting from the application of GROUP can be decrypted into unencrypted values since there is an association between them. For this, our first MET provides a function DECRYPTLABEL that maps a label into the respective unencrypted value. For instance, in Figure 3, when DECRYPTLABEL is applied over the label *Label1*, the value '2014/01' is computed and returned by DECRYPTLABEL. In summary, our first MET proposal comprises the list of functions listed as follows.

—SETUP(δ) $\rightarrow$ sk: takes a security parameter δ, and outputs a secret key sk. A secret key sk is used to encrypt unencrypted values; decrypt multivalued encrypted values; and decrypt labels into unencrypted values.
—ENCRYPT(sk, t_i) $\rightarrow$ c_i: takes a secret key sk and an unencrypted value t_i, and returns a multivalued encrypted value c_i. This function is used to encipher unencrypted values.
—DECRYPT(sk, c_i) $\rightarrow$ t_i: takes a secret key sk and a multivalued encrypted value c_i, and returns an unencrypted value t_i. This function is used to decipher multivalued encrypted values.
—GROUP(pk, c_i) $\rightarrow$ La_i: takes a public key pk (generated by a secret key sk), and a multivalued encrypted value c_i, and outputs a label La_i. GROUP is deployed as a stored procedure in an encrypted database, and is called at query runtime by the query processor for computing sorted groups of labels from multivalued encrypted values selected by queries.
—DECRYPTLABEL(sk, La_i) $\rightarrow$ t_i: takes a secret key sk and a label La_i, and returns an unencrypted value t_i. This function is used to decrypt the final results of data groupings specified in queries.

4.2 A Novel SE-based MET

Our second MET proposal defines a function ENCRYPT that associates a label with each unencrypted value, and applies a SE encryption function over a pair *(unencrypted value, label)* to generate a multivalued encrypted value. ENCRYPT guarantees that, given any unencrypted values t_i and t_j, if $t_i < t_j$, then $La_i < La_j$, where La_i and La_j are the labels associated with t_i and t_j, respectively. Also, the SE encryption used by ENCRYPT can be any SE technique found in the literature.

Figure 4 depicts labels assigned by ENCRYPT to the values of the attribute *Month*. Thus, when ENCRYPT is applied over the value '2014/01', it returns a multivalued encrypted value yielded by the SE encryption used by ENCRYPT, taking as input the pair *('2014/01', Label1)*. Inversely, our second MET defines a function DECRYPT that transforms any label into the respective unencrypted value. For instance, in Figure 4, when DECRYPT is applied over the label *Label1*, the value '2014/01' is computed and returned by DECRYPT.

Our second MET also defines a function GROUP that applies a SE decryption function to transform multivalued encrypted values into labels associated with their respective unencrypted values. Since the same labels refer to the same unencrypted values, the query processor is able to compute sorted groups

...	'2014/01'	'2014/02'	'2014/03'	...
...	Label1	Label2	Label3	...

Fig. 4. An example of labels

of labels (i.e. encrypted values). GROUP is deployed as a *stored procedure* in an encrypted database, and is invoked at runtime to be applied over multivalued encrypted values selected by queries. Thus, when a query is issued, it is transformed into another query that invokes GROUP. For example, the *projection*, *group by* and *order by* clauses specified in a SQL query as select *Month ... group by Month order by Month* are transformed into *select GR(EncMonth) as Mon ... group by Mon order by Mon*, where: *GR* is a name given to the stored procedure that implements GROUP; *EncMonth* is equivalent to the attribute *Month*, so that their values are encrypted by our second MET; and *Mon* is an alias for *GR(EncMonth)*. Then, the labels resulting from the application of GROUP can be decrypted into unencrypted values by the function DECRYPT. In summary, our second MET proposal comprises the set of functions listed as follows.

—SETUP(δ) $\rightarrow$ sk: takes a security parameter δ, and outputs a secret key sk. A secret key sk is used to encipher unencrypted values; and decrypt labels into unencrypted values.
—ENCRYPT(sk, (t_i, La_i)) $\rightarrow$ c_i: takes a secret key sk and a pair composed by an unencrypted value t_i and a label La_i, and returns a multivalued encrypted value c_i. This function is used to encipher unencrypted values.
—DECRYPT(sk, La_i) $\rightarrow$ t_i: takes a secret key sk and a label La_i, and returns an unencrypted value t_i. This function is used to decrypt the final results of data groupings specified in queries.
—GROUP(pk, c_i) $\rightarrow$ La_i: takes a public key pk (generated by a secret key sk), and a multivalued encrypted value c_i, and outputs a label La_i. GROUP is deployed as a stored procedure in an encrypted database, and is called at query runtime by the query processor to compute sorted groups of labels from multivalued encrypted values selected by queries.

5. DIMENSIONAL DATA SCHEMAS FOR ENCRYPTED DWs BASED ON METs

The proposed METs of Section 4 are used in this section to define data encryption schemas for DWs.

5.1 The Encrypted Dimensional Data Schema MV-HOM

Our first dimensional data schema proposal for encrypted DWs is called *MV-HOM*, is depicted in Figure 5, and uses our first MET proposal defined in Section 4.1 together with a HOM encryption. According to this data ecryption schema: primary and foreign keys are not encrypted; descriptive attributes are encrypted by our first MET; and measures are encrypted by our first MET and by a HOM encryption.

In this encryption schema, primary and foreign keys are kept unencrypted due to four main reasons. First, if the unencrypted values of primary and foreign keys are enciphered by METs, then the resulting multivalued encrypted values impair the processing of joins because the equality of values required by joins is not correctly evaluated by the DBMS's query processor. Second, the encryption of primary and foreign keys does not prevent the exposure of associations between data items of dimension tables and

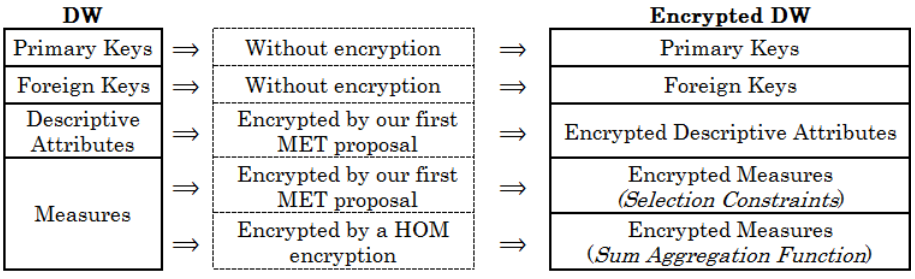

Fig. 5. The encrypted dimensional data schema MV-HOM

of fact tables being processed by joins, no mattering if these data items are encrypted or not. Third, keeping primary and foreign keys unencrypted does not reveal any semantics about them because in a DW, primary and foreign keys are often surrogate keys [Kimball and Ross 2013] composed by artificial values that do not display any business information. Fourth, the encryption of primary and foreign keys degrades the performance of join operations [Lopes et al. 2014].

Regarding the other attributes of this encryption schema (i.e. descriptive attributes and measures), the selection of their respective encryption methods was performed as follows. Analytical queries specified over a DW commonly involve the processing of selection constraints over descriptive attributes and measures; the computation of data groupings and sorting operations over descriptive attributes; and the calculation of aggregation functions (frequently based on the *sum* computation) over measures. Thus, encrypting descriptive attributes and measures by our first MET enables the processing of selection constraints, data groupings and sorting operations over multivalued encrypted values produced by this MET, while the use of a HOM encryption to encrypt measures generates multivalued encrypted values that can be useful in the calculation of sum aggregation functions.

According to this data encryption schema, each descriptive attribute in a DW yields a descriptive attribute in the encrypted DW. However, each measure in a DW yields two measures in the encrypted DW, where one is encrypted by our first MET and used in the computation of selection constraints, while the other one is coded by a HOM encryption and used in the calculation of sum aggregation functions.

Figure 6 presents the results derived from the application of the encryption schema MV-HOM to the star schema of Figure 1. In Figure 6, the attributes suffixed with *MV* represent the attributes encrypted by our first MET, and attributes with suffix *HOM* are those encrypted by a HOM encryption. Thus, using this schema, the analytical query illustrated in Section 3, and expressed in SQL syntax as *select Category, Month, SUM(Revenue) from Order, Date, Product where Month between '2014/01' and '2014/03' and Quantity < 25 and OrderProduct = ProductKey and OrderDate = DateKey group by Category, Month order by Category, Month* is mapped to *select GR(pk1, CategoryMV) as Cat, GR(pk2, MonthMV) as Mon, SUM(RevenueHOM) from EncOrder, EncDate, EncProduct where MonthMV between L'2014/01' and G'2014/03' and QuantityMV < L25 and OrderProduct = ProductKey and OrderDate = DateKey group by Cat, Mon order by Cat, Mon*, in order to be executed over the encrypted DW of Figure 6. In this query, *GR* is a name given to the stored procedure that implements the function GROUP, as defined in Section 4.1; *pk1* and *pk2* are public keys used by *GR*; the values *S'2014/01'*, *L'2014/03'* and *S25* represent integer values that are, respectively, the smallest value of the list associated with the value '2014/01', the largest value of the list related to the value '2014/03', and the smallest value of the list referred to the value 25, which were computed by the function ENCRYPT, as defined in Section 4.1.

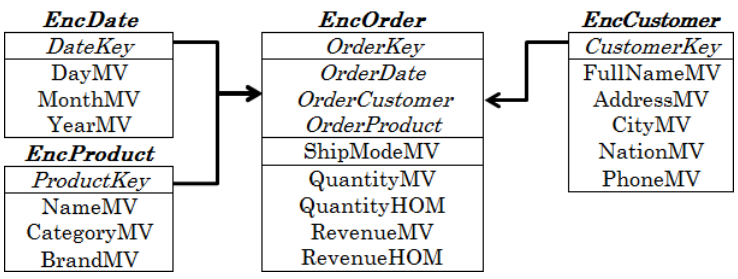

Fig. 6. An example of a schema MV-HOM

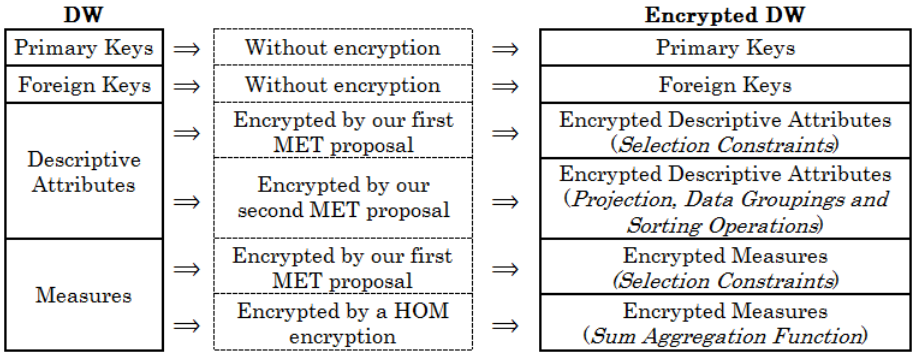

Fig. 7. The encrypted dimensional data schema MVSE-HOM

5.2 The Encrypted Dimensional Data Schema MVSE-HOM

Our second dimensional data schema proposal for encrypted DWs is called *MVSE-HOM*, is illustrated in Figure 7, and uses our two MET proposals (as defined in Section 4.1 and Section 4.2) together with a HOM encryption. According to this encryption schema, primary and foreign keys are not encrypted, as explained in Section 5.1; descriptive attributes are encrypted by our two MET proposals; and measures are encrypted by our first MET and by a HOM encryption (as stated in Section 5.1 as well). Thus, each primary key and each foreign key in a DW produce the same primary key and the same foreign key in the encrypted DW, respectively. Also, each descriptive attribute in a DW yields two descriptive attributes in the encrypted DW, where one is encrypted by our first MET and used in selection constraints, while the other one is enciphered by our second MET and used in projections, data groupings and sorting operations. Finally, each measure in a DW yields two measures in the encrypted DW, where one is encrypted by our first MET and used in selection constraints, while the other one is coded by a HOM encryption and used in the calculation of sum aggregation functions.

Figure 8 illustrates the results derived from the application of the encryption schema MVSE-HOM to the star schema of Figure 1. Attributes with suffix *MV* contain multivalued encrypted values produced by our first MET, and are used in selection constraints, while attributes with the suffix *SE* have multivalued encrypted values yielded by our second MET, and are used in projections, data groupings and sorting operations. Finally, attributes suffixed with HOM contain multivalued encrypted values generated by a HOM encryption and are used by sum aggregation functions.

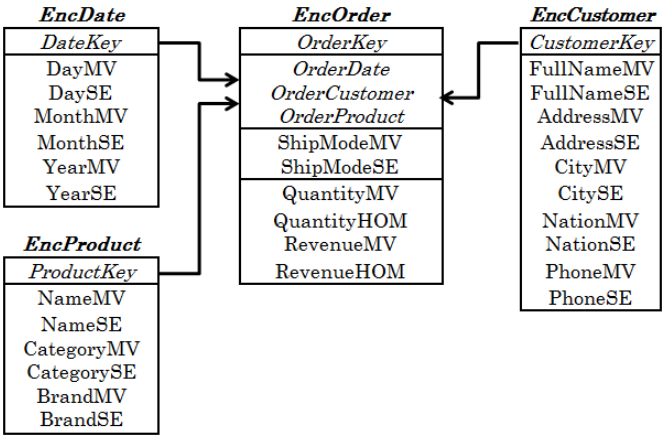

Fig. 8. An example of a schema MVSE-HOM

Using this schema, the analytical query presented in Section 3 is transformed into an analytical query expressed in SQL syntax as *select GR(key, CategorySE) as Cat, GR(key, MonthSE) as Mon, SUM(RevenueHOM) from EncOrder, EncDate, EncProduct where MonthMV between L'2014/01' and G'2014/03' and QuantityMV < L25 and OrderProduct = ProductKey and OrderDate = DateKey and group by Cat, Mon order by Cat, Mon*, to be executed over the encrypted DW of Figure 8. In this query, *GR* is the stored procedure's name that implements the function GROUP defined in Section 4.2; *key* is a public key used by *GR*; and the values *S'2014/01'*, *L'2014/03'* and *S25* represent integer values that are, respectively, the smallest value of the list associated with the value '2014/01', the largest value of the list corresponding to the value '2014/03', and the smallest value of the list defined for the value 25, which were computed by the function ENCRYPT, as defined in Section 4.1.

6. PERFORMANCE EVALUATION

We conducted experiments to evaluate the proposed encrypted dimensional data schemas MV-HOM and MVSE-HOM in terms of analytical query processing performance. In this section, we report our main results.

6.1 Workbench

We considered a distributed client-server system architecture, where the client contains an *encryption layer* that holds encryption keys, encrypts data before sending them to the encrypted DW in the server, maps analytical queries into queries to be executed over an encrypted DW, and decrypts the results returned by the execution of analytical queries over an encrypted DW. On the other hand, the server is a public cloud and stores an encrypted DW partitioned horizontally among several nodes, and provides a *scalable mechanism* to distribute the processing of analytical queries over these nodes. The client is seen as a safe environment and does not reveal information to the server about data schemas, user's analytical queries, mapping of queries and encryption keys. The server is considered as an unsafe environment that only has access to the encrypted DW.

We defined the test configurations as follows. *MV-HOM* and *MVSE-HOM* are the test configurations for the encrypted dimensional data schemas described in Sections 5.1 and 5.2 (called MV-HOM and MVSE-HOM), respectively. The HOM encryption proposed by Castelluccia et al. [2009] was used in both test configurations; while De Caro et al. [2010] implemented the SE encryption used in the *MVSE-HOM*, as stated in Section 4.2, since its code is available in the library proposed by De Caro and Iovino [2011]. Also, *NONENC* is the test configuration related to a DW with unencrypted values, and aimed at verifying the overhead caused by *MV-HOM* and *MVSE-HOM* in the processing of analytical queries. Thus, for *NONENC*, the system architecture detailed previously does not have an encryption layer. However, no other test configurations using similar encrypted dimensional data schemas were defined, because to the best of our knowledge, the literature does not report on encryption schemas that allow the execution of data groupings over multivalued encrypted values.

We executed performance tests based on the benchmark SSB [O'Neil et al. 2009] that is the standard benchmark for the performance evaluation of DW modeled according to the star schema. For each test configuration, we created a SSB dataset with about 6M records in the fact table, where *MV-HOM* and *MVSE-HOM* increased the storage cost in 44.12% and 73.97% w.r.t. *NONENC*, respectively. The client was deployed on a laptop with 2.10 GHz Pentium T4300 processor, 4 GB RAM, 5400 RPM SATA 500 GB HD, Linux Debian 6.0 32bits and JRE7. The *scalable mechanism* was deployed on the Windows Azure Platform using computers with shared CPUs, 768 MB RAM, 20 GB HD and Windows Server 2008 R2, while the SSB datasets were deployed on the Windows Azure SQL Database. The network bandwidth used between the client and the server and between the *scalable mechanism* and the SSB datasets were 6Mbps and 5Mbps, respectively. All encryption algorithms used in the tests were implemented in Java, and both versions of the function GROUP (as defined in Section 4.1 and

4.2) were implemented in Transact-SQL and deployed as a stored procedure in the Windows Azure SQL Database.

Experiments were performed so that the SSB datasets were equally partitioned among 1, 2, 4, 8 and 16 nodes. For each test configuration, each SSB query was executed 5 times, and the averages of the *client elapsed time*, *server elapsed time* and *total elapsed time* (sum of the previous ones) were collected in seconds.

6.2 Overhead of MV-HOM and MVSE-HOM w.r.t. NONENC

We investigated the overhead caused by the encrypted dimensional data schemas MV-HOM and MVSE-HOM w.r.t. a non-encrypted dimensional data schema, in the processing of all SSB queries. For this, we collected the *total elapsed time* for the test configurations *MV-HOM* (i.e. a DW encrypted according to MV-HOM), *MVSE-HOM* (i.e. a DW coded according to MVSE-HOM), and *NONENC* (i.e. an unencrypted DW). Results indicated that the overhead of both *MV-HOM* and *MVSE-HOM* reduced with the parallelization of the SSB queries' processing, although only a slighter reduction was obtained by *MVSE-HOM*. These reductions occurred because the query processor's workload for computing the SSB queries was minimized when the SSB datasets were partitioned among an increasing number of nodes. In fact, as shown in Figure 9(a), the *MV-HOM's* overhead ranged from 40.82% (with 1 node) to 10.41% (with 16 nodes) when compared to *NONENC*, while *MVSE-HOM* produced an overhead that varied from 98.95% (with 1 node) to 89.91% (with 16 nodes).

We also evaluated the overhead imposed by the computations of operations (i.e. selection constraints, joins, sum aggregation functions, data groupings and sorting operations) in the server over the data schemas MV-HOM and MVSE-HOM w.r.t. a non-encrypted DW. For this, we collected the *server elapsed time* for each proposed test configuration, as shown in Figure 9(b). Results demonstrated that the overhead caused by *MV-HOM* w.r.t. *NONENC* ranged from 43.13% (with 1 node) to 4.34% (with 16 nodes), while the *MVSE-HOM's* overhead varied from 99.11% (with 1 node) to 92.03% (with 16 nodes) with respect to *NONENC*. Therefore, based on the obtained results showed in Figures 9(a) and 9(b), we can conclude that MV-HOM did not impair the processing performance of analytical queries, since the overhead imposed by the encryption was rather reduced w.r.t. an unencrypted DW (minimum overheads of 10.41% and 4.34%, considering the *total elapsed time* and the *server elapsed time* for computing all SSB queries, respectively). However, the overhead imposed by MVSE-HOM was high when compared to an unencrypted DW (minimum overhead of 89.91%, admitting the *total elapsed time*, and minimum overhead of 92.03% assuming only the *server elapsed time*). This high MVSE-HOM's overhead is further investigated next section.

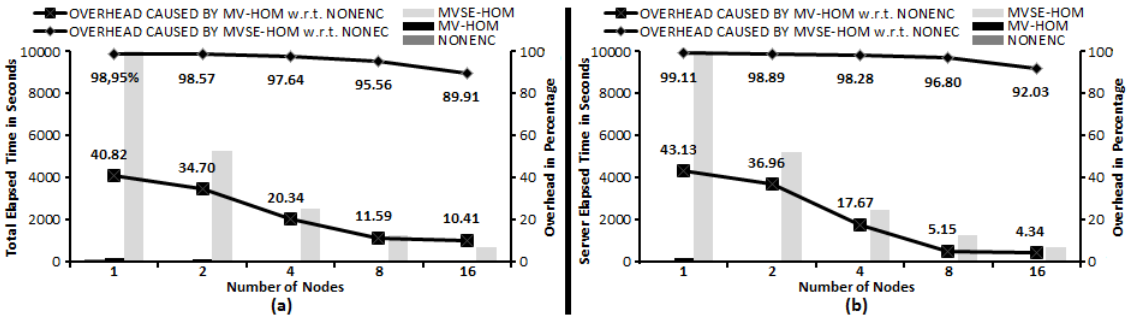

Fig. 9. Performance of MV-HOM and MVSE-HOM w.r.t. NONENC

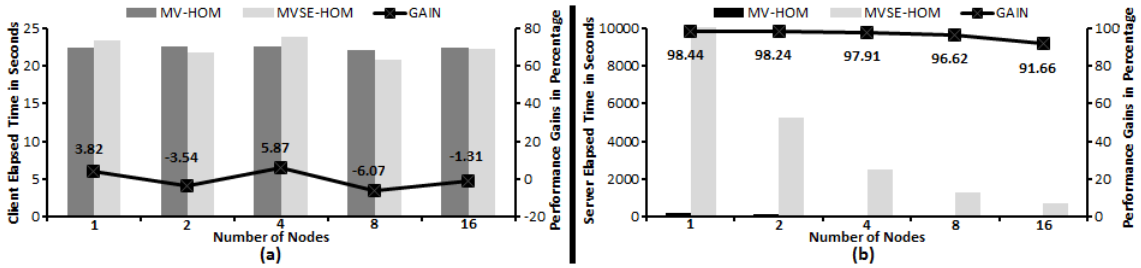

Fig. 10. Performance of MV-HOM w.r.t. MVSE-HOM

6.3 MV-HOM vs MVSE-HOM

In this experiment, we compared MV-HOM with MVSE-HOM regarding the processing performance of all SSB queries, in order to investigate the cause of the high overhead produced by MVSE-HOM w.r.t. a non-encrypted dimensional data schema. For this, we analyzed the *client elapsed time* for the test configurations *MV-HOM* and *MVSE-HOM*, and results showed similar performances. As shown in Figure 10(a), *MV-HOM* generated small performance gains w.r.t. *MVSE-HOM* of 3.82% and 5.87% with 1 and 4 nodes, respectively; and small performance losses of 3.54%, 6.07% and 1.31% with 2, 8 and 16 nodes, respectively. On the other hand, when the *server elapsed time* was considered, the results showed that *MV-HOM* obtained impressive performance gains w.r.t. *MVSE-HOM* that ranged from 91.66% (with 16 nodes) to 98.44% (with 1 nodes), as depicted in Figure 10(b). Since the selection constraints, joins, sorting operations and sum aggregation functions were executed similarly by both *MV-HOM* and *MVSE-HOM*, these results indicated that the decryption performed by *MVSE-HOM* for transforming multivalued encrypted values of SSB queries, into their corresponding labels, was time consuming. Such a decryption is done at query runtime by the function GROUP in order to compute data groupings, as stated in Section 4.2. Thus, this finding did not invalidate our second MET but showed that the proposal of De Caro et al. [2010], which implemented the SE encryption used by *MVSE-HOM* in this experiment and by the function GROUP, is not feasible to be applied over multivalued encrypted values at query runtime, since it impaired the processing of data groupings of the SSB queries.

7. CONCLUSION AND FUTURE WORK

In this article, we defined two novel METs for supporting data groupings over multivalued encrypted values. Our first MET proposal is based on multivalued order preserving encryption, and allows the processing of selection constraints, data groupings and sorting operations over multivalued encrypted values; while our second MET proposal, that is based on searchable encryption, enables the computation of data groupings and sorting operations over multivalued encrypted values. Then, we defined two novel dimensional data schemas for encrypted DWs, namely MV-HOM and MVSE-HOM, which combined our MET proposals with homomorphic encryption for allowing the execution of operations of analytical queries (i.e. selection constraints, joins, sum aggregation function, data groupings and sorting operations) over multivalued encrypted values. MV-HOM and MVSE-HOM were evaluated experimentally by enciphering DWs stored in a cloud that were partitioned among several nodes, and that enabled the parallelization of the processing of analytical queries over these nodes. Results showed that MV-HOM did not impair the performance of analytical queries because its overhead was decreased up to 10.41% (with 16 nodes) when compared to a non-encrypted dimensional data schema; while the overhead imposed by MVSE-HOM was high (89.91% with 16 nodes). This MVSE-HOM's high overhead was investigated, and results showed that the proposal of De Caro et al. [2010] used to implement the SE encryption and applied by MVSE-HOM, was time consuming during the processing of data groupings over multivalued encrypted values at query runtime.

As future work, we intend to formalize a security analysis of our MET proposals. We also intend to define the threat model for DWs encrypted according to MV-HOM and MVSE-HOM, in order to determine the types of attacks that are possible over them. Also, we are planning to implement other SE encryption techniques to be used by MVSE-HOM for computing data groupings over multivalued encrypted values. In addition, we intend to investigate the feasibility of using MV-HOM and MVSE-HOM in the computation of other types of operations (e.g. having operations) and aggregation functions (e.g. average, minimum and maximum values, and standard deviation) that may be defined in analytical queries over multivalued encrypted values. Finally, an experimental evaluation of MV-HOM and MVSE-HOM with increasing data volumes is another future work.

REFERENCES

Boneh, D., Crescenzo, G. D., Ostrovsky, R., and Persiano, G. Public Key Encryption with Keyword Search. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*. Interlaken, Switzerland, pp. 506–522, 2004.

Boneh, D., Gentry, C., Halevi, S., Wang, F., and Wu, D. J. Private Database Queries Using Somewhat Homomorphic Encryption. In *Proceedings of the International Conference on Applied Cryptography and Network Security*. Banff, Canada, pp. 102–118, 2013.

Boneh, D. and Waters, B. Conjunctive, Subset, and Range Queries on Encrypted Data. In *Proceedings of the Conference on Theory of Cryptography*. Amsterdam, The Netherlands, pp. 535–554, 2007.

Castelluccia, C., Chan, A. C.-F., Mykletun, E., and Tsudik, G. Efficient and Provably Secure Aggregation of Encrypted Data in Wireless Sensor Networks. *ACM Transactions on Sensor Networks* 5 (3): 20:1–20:36, 2009.

Chen, K., Kavuluru, R., and Guo, S. RASP: efficient Multidimensional Range Query on Attack-resilient Encrypted Databases. In *Proceedings of the ACM Conference on Data and Application Security and Privacy*. San Antonio, USA, pp. 249–260, 2011.

De Caro, A. and Iovino, V. jPBC: java Pairing-based Cryptography. In *Proceedings of the IEEE Symposium on Computers and Communications*. Kerkyra, Greece, pp. 850–855, 2011.

De Caro, A., Iovino, V., and Persiano, G. Fully Secure Anonymous HIBE and Secret-key Anonymous IBE with Short Ciphertexts. In *Proceedings of the International Conference on Pairing-based Cryptography*. Yamanaka, Japan, pp. 347–366, 2010.

Gahi, Y., Guennoun, M., and El-Khatib, K. A Secure Database System using Homomorphic Encryption Schemes. In *Proceedings of the International Conference on Advances in Databases, Knowledge, and Data Applications*. St. Maarten, The Netherlands Antilles, pp. 54–58, 2011.

Ge, T. and Zdonik, S. Answering Aggregation Queries in a Secure System Model. In *Proceedings of the International Conference on Very Large Data Bases*. Vienna, Austria, pp. 519–530, 2007.

Gentry, C. Computing Arbitrary Functions of Encrypted Data. *Communications of the ACM* 53 (3): 97–105, 2010.

Hore, B., Mehrotra, S., Canim, M., and Kantarcioglu, M. Secure Multidimensional Range Queries over Outsourced Data. *The VLDB Journal* 21 (3): 333–358, 2012.

Kadhem, H., Amagasa, T., and Kitagawa, H. MV-OPES: multivalued-Order Preserving Encryption Scheme: a Novel Scheme for Encrypting Integer Value to Many Different Values. *IEICE Transactions on Information and Systems* E93.D (9): 2520–2533, 2010.

Kadhem, H., Amagasa, T., and Kitagawa, H. Optimization Techniques for Range Queries in the Multivalued-partial Order Preserving Encryption Scheme. In A. Fred, J. L. G. Dietz, K. Liu, and J. Filipe (Eds.), *Knowledge Discovery, Knowledge Engineering and Knowledge Management*. Springer, Berlin, pp. 338–353, 2013.

Kimball, R. and Ross, M. *The Data Warehouse Toolkit: the Definitive Guide to Dimensional Modeling*. John Wiley & Sons, 2013.

Lauter, K., Naehrig, M., and Vaikuntanathan, V. Can Homomorphic Encryption be Practical? In *Proceedings of the ACM Workshop on Cloud Computing Security*. New York, USA, pp. 113–124, 2011.

Liu, D. Securing Outsourced Databases in the Cloud. In S. Nepal and M. Pathan (Eds.), *Security, Privacy and Trust in Cloud Systems*. Springer, Berlin, pp. 259–282, 2014.

Liu, D. and Wang, S. Programmable Order-Preserving Secure Index for Encrypted Database Query. In *Proceedings of the IEEE International Conference on Cloud Computing*. Honolulu, USA, pp. 502–509, 2012.

Lopes, C. C., Times, V. C., Matwin, S., Ciferri, R. R., and Ciferri, C. D. A. Processing OLAP Queries over an Encrypted Data Warehouse Stored in the Cloud. In *Proceedings of the International Conference on Data Warehousing and Knowledge Discovery*. pp. 195–207, 2014.

O'Neil, P., O'Neil, B., and Chen, X. The Star Schema Benchmark. Online Publication of Database Generation Program. Available at `http://www.cs.umb.edu/~poneil/StarSchemaB.pdf`, 2009.

Popa, R. A., Redfield, C. M. S., Zeldovich, N., and Balakrishnan, H. CryptDB: processing Queries on an Encrypted Database. *Communications of the ACM* 55 (9): 103–111, 2012.

Shi, E., Bethencourt, J., Chan, T. H., Song, D., and Perrig, A. Multi-Dimensional Range Query over Encrypted Data. In *Proceedings of the IEEE Symposium on Security and Privacy*. Oakland, USA, pp. 350–364, 2007.

Suciu, D. SQL on an Encrypted Database: technical Perspective. *Communications of the ACM* 55 (9): 102–102, 2012.

Tu, S., Kaashoek, M. F., Madden, S., and Zeldovich, N. Processing Analytical Queries over Encrypted Data. In *Proceedings of the International Conference on Very Large Data Bases*. Trento, Italy, pp. 289–300, 2013.

Vimercati, S. D. C. D., Foresti, S., Jajodia, S., Paraboschi, S., and Samarati, P. Encryption Policies for Regulating Access to Outsourced Data. *ACM Transactions on Database Systems* 35 (2): 12:1–12:46, 2010.

Wang, S., Agrawal, D., and Abbadi, A. E. Is Homomorphic Encryption the Holy Grail for Database Queries on Encrypted Data? Tech. rep., Department of Computer Science, University of California, Santa Barbara, USA, 2012.