On the Hardness of Separation of Duties Problems for Cloud Databases

Ferdinand Bollwein and Lena Wiese

[1] Institute of Applied Stochastics and Operations Research, TU Clausthal
ferdinand.bollwein@tu-clausthal.de
[2] Institute of Computer Science, University of Göttingen
wiese@cs.uni-goettingen.de

Abstract. Using cloud databases puts confidential data at risk. We apply vertical fragmentation of data tables in order to obtain insensitive data fragments. These fragments can then be hosted in databases at different cloud providers. Under the assumption that the cloud providers do not communicate, we then obtain a separation of duties such that each provider is unable to recombine the original confidential data set. In this paper, we view this separation of duties as an optimization problem. We show that it is a combination of the two famous NP-hard problems bin packing and vertex coloring. We analyze the complexity of the problem in the standard case (when only confidentiality is required) and the extended case (when also utility is a requirement).

1 Introduction

Cloud databases are a convenient solution for solving data management problems. However, when outsourcing data to a cloud service, the users (the so-called data owners) transfer the control of the data to the cloud service provider. The here presented separation of duties approach aims at protecting confidentiality of data stored in cloud databases. Consistent with [1] and [8] cloud service providers are assumed to be *honest-but-curios*: the cloud database servers answer queries correctly and do not manipulate the stored data – but they try to analyze the data and the queries in order to gain as much information as possible from them. Our work is based on the so-called *keep-a-few* approach which was introduced in [8]; we extend this basic approach in several ways – in particular, by allowing more than one external server. Like in [1], we assume that cloud service providers are *non-communicating* – otherwise servers could collaborate to reestablish sensitive information from their insensitive portions of data. The non-communication assumption can be avoided by encrypting tuple identifiers or replacing them by different placeholders in every fragment.

The separation of duties approach is based on the assumption that often the association of data is sensitive, while individual values are insensitive; under this assumption confidentiality can be protected by distributing the data among multiple servers and thereby breaking these associations. However, there is of course the possibility that certain values themselves are too sensitive to be exposed to

a cloud database provider. For this problem, there are basically two possible solutions. Either the sensitive values are encrypted before storing them in the cloud database – or the sensitive data are not outsourced at all and instead stored locally at the user's site (called the *owner* site). While the encryption approach is certainly more beneficial for the user from the storage consumption point of view (no local storage at the owner site is needed), there is only a limited possibility to perform queries on the encrypted data; encryption also involves a non-negligible overhead when encrypting and decrypting the data. A comparison of *property-preserving* encryption schemes (that support sorting and range queries on encrypted data as well as searching for encrypted keywords) is given in [21] including an assessment of the encryption overhead.

We want to emphasize the point that we are addressing the problem of outsourcing data *storage* into the cloud so that the data have to retain their original quality and accuracy. This is opposed to data *publishing* approaches, that – in order to achieve privacy-preserving statistical evaluations – distort or modify data so that the original data set is not recoverable without additional metadata that could undo the distortion or modification. Prominent approaches that include data distortion are k-anonymity [19, 20] and differential privacy [14].

To summarize our contributions, we use vertical fragmentation as a technique to protect data confidentiality from honest-but-curious cloud database servers. Consistent with related work, the confidentiality requirements are modeled as subsets of columns of the individual relations – the so-called confidentiality constraints. The resulting fragments are linkable by a common attribute but it is assumed that they are stored on separate non-communicating servers. The problem of finding such fragmentations is modeled as a mathematical optimization problem and it is one of the main objectives to minimize the number of involved servers. In this paper we extend the work in [5, 4]: we analyze the theoretical complexity of the standard separation of duties problem (that enforces confidentiality constraints) by polynomial reduction of the NP-hard problem vertex coloring. Moreover, extended requirements (visibility constraints and closeness constraints) are introduced to improve the utility of the resulting fragmentations and to enable efficient query processing. Those constraints are modeled as soft constraints – in contrast to the confidentiality requirements. We also show NP-hardness of this *extended* separation of duties problem.

Organization of the article. Related work is presented in Section 2. Section 3 introduces the main notions used in this article. Section 4 analyzes the standard variant of separation of duties under confidentiality constraints. Section 5 treats the more involved case of visibility as well as closeness constraints to enforce data locality for more performance of distributed query answering. Section 6 concludes this article with suggestions for future work.

2 Related Work

The major component of our separation of duties approach is the concept of *vertical fragmentation*. This concept is part of many standard textbooks like

[18]. Basically, the term vertical fragmentation refers to the process of dividing a relation (see Section 3.1 for a formal definition of the relational data model) into smaller units called (relation) fragments. Usually, as described in [18], this is done to speed up database systems. In related work, several approaches apply vertical fragmentation and consider attribute affinity in a given workload of transactions as an optimization measure. A recent comparative evaluation of vertical fragmentation approaches is provided in [17]; however all of these approaches do not consider fragmentation as a security mechanism.

In this paper, we apply vertical fragmentation as a powerful technique to set up a confidentiality-preserving cloud database environment. Two approaches can be seen as starting points for using vertical fragmentation to achieve confidentiality in distributed databases:

- The "two can keep a secret" approach [1] considers distribution of a single table between two database servers and leverages encryption whenever necessary to maintain confidentiality of data stored at the external servers.
- The "keep a few" approach [8] ensures confidentiality by storing highly confidential data at the owner site (in a so-called "owner fragment") and only outsourcing the remaining data to an external server.

Several extensions have spawned off these two basic approaches covering different extensions like visibility constraints and dependencies [6, 7, 9–13, 2]. In particular, in [8, 13] Hypergraph Coloring is applied to show hardness of the underlying problems. We complement these approaches by explicitly allowing more than two external servers. We formalize this as an optimization problem such that the amount of necessary cloud database providers is minimized. In our problem formulation visibility and confidentiality constraints may be in conflict (a problem that we solve by treating visibility constraints as soft constraints) – and we additionally consider closeness constraints (that improve data locality).

In general our approach is applicable to databases with multiple relations; this setting was formalized in [5]. Unlike [3] where inter-table constraints involving two relations require that no combination of attributes from these relations are stored in the same fragment, we model confidentiality constraints as subsets of the whole set of attributes from all relations of the database. With this approach the inter-table constraints from [3] can be modeled as pairwise confidentiality constraints for the attributes of the involved relations but it also allows modeling more fine-grained confidentiality concerns.

3 Background and example

3.1 Relational Data Model

This work uses the notations of the *relational database model* which was first introduced by Codd in 1970 [15] and has become one of the most commonly used models in the context of databases. The main concept in the relational model is the notion of a *relation schema*. A relation schema $R(a_1, \ldots, a_n)$, consists of a *relation name* R and a finite set of *attributes* $\{a_1, \ldots, a_n\}$ with $n \geq 1$.

Each attribute a_i is associated with a specific domain of possible values which is accounted for by the expression $dom(a_i)$. Next, a *relation (instance)* r on the relation schema $R(a_1, \ldots, a_n)$, also denoted by $r(R)$, is defined as an ordered set of *n-tuples* $r = (t_1, \ldots, t_m)$ such that each *tuple* t_j is an ordered list $t_j = \langle v_1, \ldots, v_n \rangle$ of values $v_i \in dom(a_i)$ or $v_i = \mathsf{NULL}$. The NULL value is a special constant which is used whenever a certain value of the attribute is unknown or does not apply for a certain tuple. For sake of simplicity we only discuss the theory of Separation of Duties in the context of a single relation r on schema $R(A)$ for a set A of attributes. Lastly, two relational operations are introduced. Let $r = (t_1, \ldots, t_m)$ denote a relation over the relation schema $R(A)$. The projection $\pi_f(r)$ for any $f \subseteq A$ is defined as the mapping that assigns a relation r to the set of tuples:

$$\pi_f(r) \coloneqq \{t_1[f], \ldots, t_m[f]\}.$$

This corresponds to the set of tuples of r restricted to the subset $f \subseteq A$. Projection is used to obtain the resulting relation fragments from the relation instance. The second important operation is the *equi-join*. To define this operation, let r_1 and r_2 denote relations over the relation schemes $R_1(A_1)$ and $R_2(A_2)$ respectively. The *cartesian product* of r_1 and r_2 is defined as the set of tuples $r_1 \times r_2 \coloneqq \{(t_1, t_2) \,|\, t_1 \in r_1,\ t_2 \in r_2\}$. If $A_1' \subseteq A_1$ and $A_2' \subseteq A_2$ denote subsets of attributes of the relation schemes $R_1(A_1)$ and $R_2(A_2)$, the equi-join of r_1 and r_2 on A_1' and A_2' is defined as the operator that returns the set:

$$r_1 \bowtie_{A_1' = A_2'} r_2 \coloneqq \{t \in r_1 \times r_2 \,|\, t[A_1'] = t[A_2']\}$$

In the context of vertical fragmentation, the equi-join operation is used to reconstruct the original relation from the obtained fragments by including common attributes in the fragments and performing equi-joins on those attributes.

3.2 Fragmentation

When fragmenting a relation vertically, there are two main requirements. The *first* property (completeness) is that every attribute must be placed in at least one fragment. The *second* property (reconstruction) requires that it must be possible to reconstruct the original relation from the fragments. This is usually achieved by placing a set of common attributes – the *tuple identifier* – into every fragment which makes it possible to link the individual tuples of each fragment. Equi-join operations on those attributes can then be used to reconstruct the original relation. Note that, due to the non-communicating server assumption applied in this work, linkability of fragments is not a security issue. It is further worth noting that the tuple identifier is required to form a proper subset of the fragments which prohibits fragments consisting of the tuple identifier attributes only. This requirement is due to the fact that the tuple identifier's sole purpose should be to ensure the reconstruction property. There is also a *third* property (disjointness) which is often required. This property demands that every non-tuple identifier attribute is placed in exactly one vertical fragment. A *correct vertical fragmentation* of a single relation is formally defined as follows:

Definition 1 (Vertical Fragmentation, Cardinality). *Let r be a relation on the relation schema $R(A)$. Let $tid \subset A$ be the* tuple identifier *of r. A tuple $\mathbf{f} = (f_0, \ldots, f_k)$ where $f_j \subseteq A$ for all $j \in \{0, \ldots, k\}$ is called a* correct vertical fragmentation *of r if the following conditions are met:*

1. ***Completeness:*** $\bigcup_{j=0}^{k} f_j = A$
2. ***Reconstruction****: $tid \subset f_j$, if $f_j \neq \emptyset$*
3. ***Disjointness:*** $f_i \cap f_j \subseteq tid$ *(for $f_i \neq f_j$ and $f_i, f_j \neq \emptyset$)*

The cardinality $\mathrm{card}(\mathbf{f})$ *of a correct vertical fragmentation of r is defined as the number of nonempty fragments of $\mathbf{f}$ as* $\mathrm{card}(\mathbf{f}) = \sum_{\substack{j=0 \\ f_j \neq \emptyset}}^{k} 1$. *At physical level, the* relation fragment *derived from fragment f_j is given by the projection $\pi_{f_j}(r)$.*

A fragmentation that satisfies the completeness and the reconstruction but not necessarily the disjointness property is called a *lossless fragmentation* of r.

3.3 Motivating Example

This section provides a motivating example to illustrate the ideas behind the separation of duties approach. Similar to [8], a hospital environment is considered that stores the patients' medical records in a relation as illustrated in Table 1; the patient identifiers (called PID) act as tuple identifiers.

PID	Name	DoB	ZIP	Diagnosis	Doctor
1	J. Doe	07.01.1986	12345	Flu	H. Bloggs
2	W. Lee	12.08.1974	23456	Broken Leg	G. Douglas
3	F. Jones	05.09.1963	23456	Asthma	H. Bloggs
4	G. Miller	10.02.1982	12345	Cough	H. Bloggs

Table 1. Database table storing medical records

Clearly, storing such data in a cloud database and exposing it to the provider violates the patients' privacy. Hence, this is definitely not an option for the hospital. However, it is only the association of the attributes which makes this relation problematic. This observation encourages the idea that it is possible to vertically divide the relation into multiple insensitive fragments which can be distributed among different cloud databases. The hospital develops the following confidentiality constraints to protect their patients' identities:

- The patients' names must not be stored in plaintext by an untrusted server.
- The date of birth and the ZIP-Code leak too much information about a patient's identity; they must not be stored by a single untrusted server.

A vertical fragmentation consisting of two server fragments and one owner fragment is considered confidentiality-preserving by the hospital. Both server frag-

f_0:

PID	Name
1	J. Doe
2	W. Lee
3	F. Jones
4	G. Miller

f_1:

PID	DoB
1	07.01.1986
2	12.08.1974
3	05.09.1963
4	10.02.1982

f_2:

PID	ZIP	Diagnosis	Doctor
1	12345	Flu	H. Bloggs
2	23456	Broken Leg	G. Douglas
3	23456	Asthma	H. Bloggs
4	12345	Cough	H. Bloggs

Table 2. One owner fragment f_0 and two server fragments f_1 and f_2

ments f_1 and f_2 (see Table 2) are insensitive and can therefore be placed on two separate cloud database providers. As long as the respective database providers are not collaborating to reestablish the sensitive associations, the confidentiality requirements imposed by the hospital are met. Because the names of the patients of a hospital are considered to be sensitive on their owns, fragment f_0 (see, Table 2) has to be treated differently. Basically, there are two options: Encrypting the names before outsourcing – or storing them locally in an owner fragment and not in a cloud database. In this work, the second option is chosen: no encryption – limiting the execution of queries involving patients' names – is necessary.

3.4 Data Distribution as Optimization Problems

We will later on combine two different NP-hard problems to obtain our Separation of Duties problem formulation. We now introduce the two underlying problems: Bin Packing and Vertex Coloring.

When outsourcing the data to cloud databases, it might be required that certain capacities in terms of storage space are not exceeded – otherwise the cloud provider could for example charge more usage fees. A famous NP-hard problem considering capacities is Bin Packing; this well-known NP-hard problem is for example stated in [16] (we adapt the notation to our purposes):

Definition 2 (Bin Packing). *Given a set $B = \{b_1, \ldots, b_k\}$ of bins (each with a maximum capacity W_j) and a set $O = \{o_1, \ldots, o_n\}$ of objects (each with a capacity consumption w_i), find the minimum number K such that all objects in O are placed in some bin, the set of used bins $U \subseteq B$ is of cardinality K (that is, $|U| = K$) and the capacities are not exceeded (that is, for each $b_j \in U$ it holds that $\sum_{o_i \in b_j} w_i \leq W_j$).*

The data distribution problem with capacity constraints is basically a Bin Packing Problem (BPP) in the following sense:

- k servers correspond to k bins
- each server b_j has a maximum capacity W_j
- n attributes correspond to n objects
- each attribute has a capacity consumption w_i
- attributes have to be placed into a minimum number of servers K without exceeding the maximum capacities W_j

On the other hand (to later on ensure confidentiality) we have to express that certain attributes should *not* be placed on the same server. This can be achieved by a graph coloring problem – more precisely Vertex Coloring; this well-known NP-hard problem is also stated in [16] (we again slightly adapt the notation):

Definition 3 (Vertex Coloring). *Given an undirected graph $\mathcal{G} = (N, E)$ consisting of nodes $N = \{n_1, \ldots, n_n\}$ and edges $E \subseteq N \times N$ with $n_i \neq n_{i'}$ for all $(n_i, n_{i'}) \in E$, find the minimum number K such that there exists a K-coloring $\varphi : N \longrightarrow \{1, \ldots, K\}$ that satisfies $\varphi(n_i) \neq \varphi(n_{i'})$ for every edge $(n_i, n_{i'}) \in E$.*

The data distribution problem with pairwise confidentiality constraints is basically a Vertex Coloring Problem (VCP) in the following sense:

- k available servers correspond to the amount k of available colors
- n attributes correspond to n nodes
- if a confidentiality constraint requires that two attributes a_i and $a_{i'}$ should be assigned to different fragments, there should be an edge $(n_i, n_{i'}) \in E$ between the two nodes n_i and $n_{i'}$ corresponding to the attributes; in effect, the two nodes will be colored with two different colors which corresponds to placing them on different servers.
- finding the minimum number K of occupied servers corresponds to finding the minimum number of colors.

In this paper we extend these basic data distribution problems into separation of duties problems that take confidentiality constraints into account and furthermore consider several additional optimization goals and settings. In particular, we consider the capacities and weights in the bin packing problem as one component of our overall optimization problem while confidentiality constraints are enforced by the constraints of the vertex coloring problem.

4 Standard Separation of Duties Problem

We now move on to the formal specification of our Separation of Duties problems. The security requirements are at attribute level, i.e. certain attributes or combinations of attributes are considered sensitive and must not be stored by a single untrusted database server. This can – consistently with related work [1] – be modeled with the notion of *confidentiality constraints*.

Definition 4 (Confidentiality Constraints). *Let $R(A)$ be a relation schema over the set of attributes A. A confidentiality constraint on $R(A)$ is defined by a subset of attributes $c \subseteq A$ with $c \neq \emptyset$. Two types of constraints are distinguished:*

- ***Singleton Constraint:*** *A singleton constraint is a confidentiality constraint c with $|c| = 1$. This means that the confidentiality constraint consists of a single sensitive attribute which should not be exposed to any untrusted server.*
- ***Association Constraint:*** *An association constraint satisfies $|c| > 1$. This means that a server is not allowed to store the combination of attributes contained in c. However, any proper subset of c may be revealed.*

Because attributes contained in a singleton constraint are not allowed to be accessed by an untrusted server, they cannot be outsourced in plaintext at all. Hence, because our approach works without encryption, those attributes have to be stored locally at the owner site. On the other hand, association constraints can be satisfied by distributing the respective attributes among two or more database servers. More precisely, a correct vertical fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ has to be found in which one fragment stores all the attributes contained in singleton constraints and all other fragments are not a superset of any confidentiality constraint. As a common convention throughout the rest of this work, fragment f_0 will always denote the *owner fragment* which stores all the attributes contained in singleton constraints. This fragment is stored by a local, trusted database. The other fragments $f_1, \ldots, f_k$ denote the *server fragments* and each of those is stored by a different untrusted database server. This leads to the formal definition of a *confidentiality-preserving vertical fragmentation*:

Definition 5 (Confidentiality-preserving Vertical Fragmentation). *For a relation r on the relation schema $R(A)$ and a set of confidentiality constraints C, a correct vertical fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ is* confidentiality-preserving *with respect to C if $c \nsubseteq f_j$, for all $c \in C$ and $j \geq 1$.*

The condition requires that no attributes contained in a confidentiality constraint are simultaneously stored in a server fragment and hence, exposed to an untrusted cloud database provider. On the one hand, this ensures that no confidentiality constraint is violated for any server fragment f_j for $j \in \{1, \ldots, k\}$. On the other hand, this means that all attributes contained in singleton constraints must be placed in the owner fragment f_0.

It is necessary to introduce some reasonable restrictions to the set of confidentiality constraints. These restrictions are of theoretical nature and will not restrict its expressiveness. These requirements are summarized by the following definition of a *well-defined* set of confidentiality constraints:

Definition 6 (Well-defined Set of Confidentiality Constraints). *Given a relation r on the relation schema $R(A)$ and a designated tuple identifier $tid \subset A$. A set of confidentiality constraints C is* well-defined *if it satisfies:*

1. *For all $c, c' \in C$ with $c \neq c'$, it holds that $c \nsubseteq c'$.*
2. *For all $c \in C$, it holds that $c \cap tid = \emptyset$.*

The first condition requires that no confidentiality constraint c is a subset of another confidentiality constraint c'. By the definition of a confidentiality-preserving vertical fragmentation, the satisfaction of c' would be redundant because $c \nsubseteq f_j$ for $j \in \{1, \ldots, k\}$ implies that $c' \nsubseteq f_j$ for $j \in \{1, \ldots, k\}$ if $c \subseteq c'$. The second condition requires that the tuple identifier attributes are considered insensitive: the tuple identifier's sole purpose is to ensure the reconstruction of the fragmentation by placing it in every nonempty fragment.

Space requirements might also be an important factor for the vertically fragmented relation and the owner and the server fragments may not exceed certain storage capacities. Hence, the concept of attribute weight is introduced:

Definition 7 (Weight Function). *Let r be a relation over the relation schema $R(A)$ and let $\mathcal{P}(A)$ denote the power set of the set of attributes A. A* weight function *for r is defined as a function $w_r : \mathcal{P}(A) \longrightarrow \mathbb{R}_{\geq 0}$ that satisfies:*

- *$w_r(f) = 0$, if and only if $f = \emptyset$*
- *$w_r(f) = \sum_{a \in f} w_r(\{a\})$ for all $f \subseteq A$*

To denote the weight of a single attribute $a \in A$, the notation $w_r(a)$ is used instead of $w_r(\{a\})$.

Due to the second condition such a weight function is fully defined by the values $w_r(a)$ for all attributes $a \in A$: any subset of A is a combination of these attributes and its weight is defined by the sum of the weights of its elements.

Keeping the number of involved server as low as possible will reduce the user's costs, lower the complexity of maintaining the vertically fragmented relation and also increase the efficiency of executing queries. Therefore, in the following problem statement, the objective is to find a confidentiality-preserving correct vertical fragmentation of minimal cardinality. Additionally, the capacities of the involved servers must not be exceeded.

Definition 8 (Standard Separation of Duties Problem). *Given a relation r over the relation schema $R(A)$, a well-defined set of confidentiality constraints C, a tuple identifier tid $\subset A$, a weight function w_r, servers $S_0, \ldots, S_k$ (where S_0 denotes the owner's trusted server and $S_1, \ldots S_k$ denote the untrusted external servers) and maximum capacities $W_0, \ldots, W_k \in \mathbb{R}_{\geq 0}$, find a correct confidentiality-preserving fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ of minimal cardinality such that the capacities are not exceeded, i.e. $w_r(f_j) \leq W_j$ for all $0 \leq j \leq k$.*

One should note that in this general formulation the owner fragment can possibly contain all of the attributes if W_0 is sufficiently large. Yet, the purpose of the owner fragment is to store the attributes contained in singleton constraints. Therefore, in a variation of this problem, the capacity of the owner fragment is chosen such that it cannot hold any attribute other than the most sensitive ones and, of course, the tuple identifier; all other attributes are actually distributed among the server fragments. If there are singleton constraints, this can be achieved by choosing W_0 as $w_r(\text{tid}) + \sum_{c \in C : |c| = 1} w_r(c)$.

4.1 Complexity Analysis

In this section, the complexity of the Standard Separation of Duties Problem is analyzed. The problem can be viewed as a combination of two famous NP-hard problems, the Bin Packing Problem due to the capacity constraints and the Vertex Coloring Problem due to the confidentiality constraints. Both problems can easily be modeled as a Separation of Duties Problem to prove NP-hardness of the latter. In real life scenarios however, the capacity constraints might often be less important because cloud storages can generally be enlarged on demand. Therefore, the proof is based on the Vertex Coloring Problem. The following theorem is proven by a polynomial reduction of an instance of the Vertex Coloring

Problem to an instance of the Standard Separation of Duties Problem. For simplicity, the former is denoted by VC and the latter by SSoD. The proof proceeds by finding a fragmentation of minimal cardinality K for SSoD which can then define a coloring for VC; lastly, we show that if K is a minimal fragmentation, there can be no coloring for VC with $K' < K$ colors.

Theorem 1. *The Standard Separation of Duties Problem is NP-hard.*

Proof. Let VC be defined by a graph $\mathcal{G} = (N, E)$ with nodes $N = \{n_1, \ldots, n_k\}$ and edges $E = \{e_1, \ldots, e_m\} \subseteq N \times N$. Without loss of generality, it is assumed that if E contains the edge $(n_i, n_{i'})$, it does not contain the equivalent edge $(n_{i'}, n_i)$. Then, SSoD is defined as follows for all $a \in A$ and $j = 1, \ldots, k$:

- For every node $n_i \in N$, an attribute a_{n_i} is defined. Additionally, an artificial tuple identifier tid $\coloneqq \{a_{\text{tid}}\}$ is introduced. Overall, the set of attributes is therefore defined by $A \coloneqq \text{tid} \cup \{a_{n_i} \mid n_i \in N\}$.
- $R(A)$ is a relation schema over A and r is a relation on $R(A)$.
- The weight function $w_r : \mathcal{P}(A) \longrightarrow \mathbb{R}_{\geq 0}$ is defined by $w_r(\emptyset) \coloneqq 0$, $w_r(a) \coloneqq 1$.
- There is an owner server S_0 and external servers S_j for each node n_j.
- The capacity of the owner's server W_0 equals zero and the servers' capacities are all large enough to potentially hold all the attributes, i.e. $W_j \coloneqq w_r(A)$.
- For every edge $(n_i, n_{i'}) \in E$, a confidentiality constraint $\{a_{n_i}, a_{n_{i'}}\} \subseteq A \times A$ is introduced. The set of confidentiality constraints is hence defined by $C \coloneqq \left\{\left\{a_{n_i}, a_{n_{i'}}\right\} \subseteq A \times A \mid (n_i, n_{i'}) \in E\right\}$. This set is well-defined because $a_{\text{tid}} \cap c = \emptyset$ for all $c \in C$ and it is assumed that if E contains $(n_i, n_{i'})$, it does not contain $(n_{i'}, n_i)$ which ensures that $c \not\subseteq c'$ for all $c, c' \in C$.

By definition, for both instances of VC and SSoD a feasible solution exists: In the former, there always exists a k-coloring which assigns each of the k nodes to a different color. In the latter, the number of servers is the same as the number of non-tuple-identifier attributes and each servers' capacity is sufficiently large to hold such an attribute together with the tuple identifier. Hence, the correct vertical fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ with $f_0 = \emptyset$ and $f_j = \{a_{\text{tid}}, a_{n_j}\}$ for $j \in \{1, \ldots, k\}$ satisfies the capacity constraints and is thus a feasible solution.
Hence, a correct privacy-preserving vertical fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ of *minimal* cardinality $\text{card}(\mathbf{f}) \coloneqq K$ of SSoD can be assumed. Without loss of generality, for this fragmentation it can be assumed that $f_0 = \emptyset$, $f_1, \ldots, f_K \neq \emptyset$ for $K \leq k$ and $f_{K+1}, \ldots, f_k = \emptyset$ because all the servers' capacities are the same and hence, the server fragments can be permuted such that the fragmentation satisfies this property. In the definition of SSoD, every non-tuple-identifier attribute $a_{n_i} \in A$ corresponds to a node n_i which allows the definition of a K-coloring $\varphi : N \longrightarrow \{1, \ldots, K\}$ from that fragmentation as $\varphi(n_i) \mapsto j$, if $a_{n_i} \in f_j$: if the attribute n_i is contained in fragment $f_j \in \mathbf{f}$, then the color j is chosen. This coloring is well-defined due to the disjointness and the completeness property of $\mathbf{f}$. More precisely, because each a_{n_i} is contained in exactly one server fragment f_j, each n_i is assigned exactly one color j. Because the confidentiality constraints are derived from the edges of the graph, it is not possible that attributes a_{n_i} and

$a_{n_{i'}}$ are in the same fragment if there exists an edge $(n_i, n_{i'}) \in E$. Therefore, no adjacent nodes are assigned the same color. This coloring uses $\text{card}(\mathbf{f}) = K$ colors – one for each nonempty fragment.
It remains to show that the numbers of colors used is in fact minimal. For that, it is assumed that there exists a K'-coloring ϕ of $\mathcal{G}$ with $K' < K$. Then, for every color $j' \in \{1, \ldots, K'\}$ in the image of ϕ, define the set $f'_{j'}$ as follows:

$$f'_{j'} \coloneqq \{a_{n_i} \in A \mid n_i \in N \text{ and } \phi(n_i) = j'\} \cup \text{tid}$$

Additionally, $f'_0 \coloneqq \emptyset$ and $f'_{K'+1}, \ldots, f'_k \coloneqq \emptyset$ is defined. Then, $\mathbf{f'} = (f'_0, \ldots, f'_k)$ forms a confidentiality-preserving correct vertical fragmentation of cardinality K': Every node n_i is assigned exactly one color $j' \in \{1, \ldots, K'\}$ and hence, every attribute a_{n_i} is contained in exactly one fragment, namely $f_{j'}$. Thus, $\mathbf{f'}$ satisfies the completeness and the disjointness property. Additionally, as the tuple identifier is included in every nonempty fragment, those fragments form a correct vertical fragmentation of r. The confidentiality constraints of SSoD are derived from the edges of $\mathcal{G}$ and therefore, there is a confidentiality constraint $\{a_{n_i}, a_{n_{i'}}\}$ if $(n_i, n_{i'}) \in E$. Moreover, the coloring satisfies $\phi(n_i) \neq \phi(n_{i'})$ if $(n_i, n_{i'}) \in E$ which means that attributes a_{n_i} and $a_{n_{i'}}$ are placed in different fragments and therefore, the corresponding confidentiality constraint is satisfied. This proves that the fragmentation is indeed confidentiality-preserving. This, however, contradicts the assumption that the cardinality K of $\mathbf{f}$ is minimal. This means, that the previously defined coloring φ is in fact minimal and a solution to the vertex coloring problem which concludes the proof of the theorem.

5 Extended Separation of Duties Problem

While the problem formulation for the Standard Separation of Duties Problem is suitable to preserve confidentiality, several enhancements will now be presented to make it applicable for practical purposes. These enhancements are mainly concerned with the distribution of the attributes to allow efficient query processing. In many scenarios, it is desirable that certain combinations of attributes are stored by a single server or in other words, these combinations are visible on a single server, because they are often queried together. This can be accounted for with the notion of *visibility constraints*:

Definition 9 (Visibility Constraint). *Let $R(A)$ denote a relation schema over the set of attributes A and let r be a relation over $R(A)$. A* visibility constraint *over $R(A)$ is a subset of attributes $v \subseteq A$. A fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ satisfies v if there exists $0 \leq j \leq k$ such that $v \subseteq f_j$. In this case, define $\text{sat}_v(\mathbf{f}) \coloneqq 1$ and $\text{sat}_v(\mathbf{f}) \coloneqq 0$ otherwise. Furthermore, for any set V of visibility constraints define $\text{sat}_V(\mathbf{f}) \coloneqq \sum_{v \in V} \text{sat}_v(\mathbf{f})$ to count the number of satisfied visibility constraints.*

In contrast to confidentiality constraints, the fulfillment of visibility constraints is not mandatory, i.e. confidentiality constraints are *hard constraints* while visibility constraints are *soft constraints*. While we require the resulting fragmentation

to satisfy the completeness property, breaking the disjointness property can help increase the number of satisfied visibility constraints. Hence, in the upcoming problem definition only a lossless fragmentation will be required.

In case a visibility constraint cannot be satisfied (because otherwise a confidentiality constraint will be violated), the distribution of the visibility constraint attributes is arbitrary.

Therefore, it is reasonable to provide constraints to ensure that certain attributes be distributed among as few servers as possible. Moreover, as in the following problem statement a lossless fragmentation will be required, those constraints can also be used to limit the number of copies of any individual attribute. We introduced so-called *closeness constraints* in [4]:

Definition 10 (Closeness Constraint [4]). *Let r be a relation over relation schema $R(A)$. A closeness constraint over $R(A)$ is a subset of attributes $\gamma \subseteq A$. Let $\mathbf{f} = (f_0, \ldots, f_k)$ be a correct/lossless vertical fragmentation of r, the distribution $\operatorname{dist}_\gamma(\mathbf{f})$ of γ is defined as the number of fragments that contain one of the attributes in γ: $\operatorname{dist}_\gamma(\mathbf{f}) \coloneqq \sum_{f_j \cap \gamma \neq \emptyset} 1$ (for $j = 0 \ldots k$). Moreover, for any set Γ of closeness constraints, the distribution $\operatorname{dist}_\Gamma(\mathbf{f})$ is defined as the sum of distributions of $\gamma \in \Gamma$: $\operatorname{dist}_\Gamma(\mathbf{f}) \coloneqq \sum_{\gamma \in \Gamma} \operatorname{dist}_\gamma(\mathbf{f})$.*

The minimization of the distribution of the closeness constraints is the third goal in the following problem formulation. However, minimizing the cardinality of the fragmentation, maximizing the number of satisfied visibility constraints and minimizing the distribution of the closeness constraints are three separate, non-complementary goals. Hence, a balance has to be found between them. Therefore, the objective stated in the problem definition is expressed as a weighted sum of these three goals using weights α_1 (for the cardinality), α_2 (for visibility) and α_3 (for closeness). Note that satisfying the confidentiality constraints is still mandatory. The *Extended Separation of Duties Problem* is defined as follows:

Definition 11 (Extended Separation of Duties Problem). *Given a relation r over a relation schema $R(A)$, a well-defined set of confidentiality constraints C, a set of visibility constraints V, a set of closeness constraints Γ, a tuple identifier $tid \subset A$, a weight function w_r, servers $S_0, \ldots, S_k$, maximum capacities $W_0, \ldots, W_k \in \mathbb{R}_{\geq 0}$ and weights $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}_{\geq 0}$. Find a lossless confidentiality-preserving fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$ of minimal cardinality which satisfies $w_r(f_j) \leq W_j$ for all $0 \leq j \leq k$ such that the weighted sum $\alpha_1 \operatorname{card}(\mathbf{f}) - \alpha_2 \operatorname{sat}_V(\mathbf{f}) + \alpha_3 \operatorname{dist}_\Gamma(\mathbf{f})$ is minimal.*

Generally the choice of the weights α_1, α_2 and α_3 depends on the application. Yet, a reasonable choice is to assign priorities to the three different objectives. In most scenarios, the overall number of necessary servers will have the highest impact on the utility and therefore, minimizing it should have the highest priority. The satisfaction of visibility constraints has the second highest priority. Finally, among those solutions, the distribution of the closeness constraints should be minimized. Under the assumption that $|V| > 0$ and $|\Gamma| > 0$, one possible solution is given by $\alpha_1 = 1$, $\alpha_2 = \frac{0.9}{2|V|}$ and $\alpha_3 = \frac{0.87}{2(k+1)|V||\Gamma|}$.

5.1 Complexity Analysis

Next, NP-hardness of the Extended Separation of Duties Problem is shown. To accomplish this, the similarity to the standard version is used.

Theorem 2. *The Extended Separation of Duties Problem is NP-hard.*

Proof. The proof is by reducing an instance of the Standard Separation of Duties Problem, denoted by SSoD, on an instance of the Extended Separation of Duties Problem, denoted by ESoD. This is done by canonically adopting the provided parameters and additionally defining the set of visibility constraints $V \coloneqq \emptyset$ and the set of closeness constraints $\Gamma \coloneqq \emptyset$. Formally, let SSoD be defined by a relation r over a relation schema $R(A)$, a tuple identifier tid $\subset A$, a set of well-defined confidentiality constraints C, a weight function w_r, servers $S_0, \ldots, S_k$ and maximum capacities $W_0, \ldots, W_k \in \mathbb{R}_{\geq 0}$. Then, ESoD is defined as follows:

- The relation r over the relation schema $R(A)$
- The tuple identifier tid
- The weight function $w_r : \mathcal{P}(A) \longrightarrow \mathbb{R}_{\geq 0}$
- The servers S_j for $0 \leq j \leq k$
- The server capacities W_j for $0 \leq j \leq k$
- The set of confidentiality constraints C
- A set of visibility constraints $V \coloneqq \emptyset$
- A set of closeness constraints $\Gamma \coloneqq \emptyset$
- Weights $\alpha_1 \coloneqq 1$, $\alpha_2 \coloneqq 1$ and $\alpha_3 \coloneqq 1$

Let a solution of ESoD be given by a lossless confidentiality-preserving fragmentation $\mathbf{f} = (f_0, \ldots, f_k)$. As V and Γ are empty, this fragmentation must be of minimal cardinality. For SSoD however, a correct fragmentation is required and therefore, to establish the disjointness property, duplicate attributes must be eliminated from some fragments in $\mathbf{f}$ to obtain a correct fragmentation $\mathbf{f}'$. This can be achieved in polynomial time. The correct fragmentation $\mathbf{f}'$ then solves both the standard and the extended version of the Separation of Duties Problem. To conclude the proof, the case that one of the instances is not solvable must be discussed. Due to the fact that every correct vertical fragmentation is also lossless and that every lossless fragmentation can be transformed into a correct one by removing duplicate attributes, it is clear that there exists a correct vertical fragmentation of r if and only if there exists a lossless vertical fragmentation of r. Hence, SSoD is solvable if and only if ESoD is also solvable.

6 Conclusion and Future Work

In this work we have presented a practical approach for preserving confidentiality in cloud databases which does not require encryption. Our separation of duties approach is based on the observation that by vertical fragmentation and by distribution of fragments among multiple non-communicating cloud database servers, sensitive associations among columns can be broken such that each of

these servers only stores an insensitive portion of the database. To model the confidentiality concerns, confidentiality constraints were introduced. Visibility constraints and closeness constraints were introduced to increase the utility of the resulting vertically fragmented database. The problem of finding such confidentiality-preserving vertical fragmentations was shown to be NP-hard.

Our approach was studied in this paper for a single database relation. However it more generally applies as well to databases consisting of many relations as studied previously in [5] to make the theory applicable in practical scenarios. Moreover, because certain combinations of attributes often reveal sensitive information about others, data dependencies can be introduced, too – this setting was also considered in [5]. Together, the confidentiality constraints and data dependencies are capable of expressing a wide range of confidentiality concerns that appear in the context of cloud databases.

One possibility to expand this work is to develop heuristics for solving the Separation of Duties Problem. The evaluation in [5] has shown that modern ILP solvers are capable of quickly finding confidentiality-preserving fragmentations of minimal cardinality. However, the introduction of visibility constraints can increase the time for finding an optimal solution significantly. Therefore, heuristics can be beneficial in situations where a long runtime is expected. Furthermore, as sensitive associations could not only occur between columns but also between rows of a database, another interesting extension of this work is to additionally explore horizontal fragmentation [22] which means that database tables are fragmented by rows. In combination with vertical fragmentation this leads to the problem of finding confidentiality-preserving hybrid fragmentations.

Last but not least, it might be worthwhile to study separation of duties also for non-relational data models (as for example surveyed in [23]).

References

1. Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Motwani, R., Srivastava, U., Thomas, D., Xu, Y.: Two can keep a secret: A distributed architecture for secure database services. In: The Second Biennial Conference on Innovative Data Systems Research (CIDR 2005) (2005)
2. Biskup, J., Preuß, M., Wiese, L.: On the inference-proofness of database fragmentation satisfying confidentiality constraints. In: International Conference on Information Security. pp. 246–261. Springer (2011)
3. Bkakria, A., Cuppens, F., Cuppens-Boulahia, N., Fernandez, J.M., Gross-Amblard, D.: Preserving multi-relational outsourced databases confidentiality using fragmentation and encryption. JoWUA 4(2), 39–62 (2013)
4. Bollwein, F., Wiese, L.: Closeness constraints for separation of duties in cloud databases as an optimization problem. In: British International Conference on Databases. pp. 133–145. Springer (2017)
5. Bollwein, F., Wiese, L.: Separation of duties for multiple relations in cloud databases as an optimization problem. In: Proceedings of the 21st International Database Engineering & Applications Symposium. pp. 98–107. ACM (2017)
6. Ciriani, V., De Capitani Di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Fragmentation and encryption to enforce privacy in data storage. In:

European Symposium on Research in Computer Security. pp. 171–186. Springer (2007)
7. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Fragmentation design for efficient query execution over sensitive distributed databases. In: ICDCS. pp. 32–39. IEEE Computer Society (2009)
8. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Keep a few: Outsourcing data while maintaining confidentiality. In: ESORICS. Lecture Notes in Computer Science, vol. 5789, pp. 440–455. Springer (2009)
9. Ciriani, V., De Capitani Di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Combining fragmentation and encryption to protect privacy in data storage. ACM Transactions on Information and System Security (TISSEC) 13(3), 22 (2010)
10. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Selective data outsourcing for enforcing privacy. Journal of Computer Security 19(3), 531–566 (2011)
11. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Livraga, G., Samarati, P.: An OBDD approach to enforce confidentiality and visibility constraints in data publishing. Journal of Computer Security 20(5), 463–508 (2012)
12. De Capitani di Vimercati, S., Erbacher, R.F., Foresti, S., Jajodia, S., Livraga, G., Samarati, P.: Encryption and fragmentation for data confidentiality in the cloud. In: Foundations of Security Analysis and Design VII, pp. 212–243. Springer (2014)
13. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Livraga, G., Paraboschi, S., Samarati, P.: Fragmentation in presence of data dependencies. IEEE Transactions on Dependable and Secure Computing 11(6), 510–523 (2014)
14. Dwork, C.: Differential privacy: A survey of results. In: International Conference on Theory and Applications of Models of Computation. pp. 1–19. Springer (2008)
15. Frank Codd, E.: A relational model of data for large shared data banks. Communications of the ACM 13(6), 377–387 (Jun 1970)
16. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1979)
17. Jindal, A., Palatinus, E., Pavlov, V., Dittrich, J.: A comparison of knives for bread slicing. Proceedings of the VLDB Endowment 6(6), 361–372 (2013)
18. Ozsu, M.T.: Principles of Distributed Database Systems. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edn. (2007)
19. Samarati, P.: Protecting respondents identities in microdata release. IEEE transactions on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
20. Sweeney, L.: k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(05), 557–570 (2002)
21. Waage, T., Wiese, L.: Property preserving encryption in NoSQL wide column stores. In: Cloud and Trusted Computing (OnTheMove Federated Conferences). Springer (2017)
22. Wiese, L.: Horizontal fragmentation for data outsourcing with formula-based confidentiality constraints. In: IWSEC. Lecture Notes in Computer Science, vol. 6434, pp. 101–116. Springer (2010)
23. Wiese, L.: Advanced Data Management for SQL, NoSQL, Cloud and Distributed Databases. DeGruyter (2015)