Privacy of Outsourced Data

Sabrina De Capitani di Vimercati and Sara Foresti

Dipartimento di Tecnologie dell'Informazione
Università degli Studi di Milano
Via Bramante 65 - 26013 Crema, Italy
firstname.lastname@unimi.it

Abstract. Data outsourced to an external storage server are usually encrypted since there is the common assumption that all data are equally sensitive. The encrypted data however cannot be efficiently queried and their selective release is not possible or require the application of specific solutions. To overcome these problems, new proposals have been recently developed, which are based on a fragmentation technique possibly combined with encryption. The main advantage of these proposals is that they limit the use of encryption, thus improving query execution efficiency. In this paper, we describe such fragmentation-based approaches focusing in particular on the different data fragmentation models proposed in the literature. We then conclude the paper with a discussion on some research directions.

1 Introduction

Data outsourcing is emerging today as a successful paradigm allowing individuals and organizations to exploit external services for storing, managing, and distributing huge collections of possibly sensitive data. Within a data outsourcing architecture, data are stored together with application front-ends at the sites of an external server who takes full charges of their management. Although publishing data on external servers may increase service availability, reducing data owners' burden of managing data, it introduces new privacy and security concerns. As a matter of fact, the outsourced data are no more under the control of their data owners and therefore their privacy as well as their integrity may be put at risk. The protection of the privacy of the data is however of paramount importance and is becoming an emerging problem as it is also testified by a number of recent regulations that require organizations to provide privacy guarantees when storing, processing, and sharing sensitive information (e.g., California Senate Bill 1386 and the Personal Data Protection code - legislative decree no. 196/2003). Existing approaches (e.g., [1–4]) for protecting the privacy of outsourced data assume that an overlying layer of encryption is applied on the data before outsourcing them, which implies that the outsourced data cannot be efficiently queried and that a selective release (i.e., different pieces of information to different parties) is either not possible or require the application of specific solutions based on two layers of encryption [5, 6]. Recently, novel proposals have been developed, where encryption is not mandatory for ensuring protection [7–9]. This introduces a paradigm shift that permits to address the protection issue with a different perspective, thus giving the possibility of designing novel models and techniques where the use of encryption is minimized or is absented. These proposals are based on the observation that often what is sensitive is the association among data more than the data per se. For instance, in a hospital the list of illnesses cured and the list of hospitalized patients could be made publicly available, since what is sensitive is the association of a specific illness to a patient. Although this association must be protected, it is not necessary to encrypt both the list of illnesses and the list of hospitalized patients; it is sufficient to prevent their joint visibility to non authorized users.

In this paper, we illustrate recent proposals for protecting outsourced data that are based on the use of fragmentation possibly combined with encryption. The remainder of this paper is organized as follows. Section 2 introduces the basic scenario and concepts on which all the fragmentation-based proposals rely on. Section 3 describes an approach based on the combination of fragmentation and encryption and where the outsourced data are stored at two non-communicating pair of servers. Section 4 presents an approach where again fragmentation and encryption are used in combination and where outsourced data can be fragmented among multiple unlinkable fragments. Section 5 illustrates a proposal that departs from encryption and where a small portion of the data is stored at the data owner side. Section 6 presents some open issues for the considered scenario. Finally, Section 7 concludes the paper.

2 Scenario and basic assumptions

We consider the problem of outsourcing data while preserving their privacy. Current approaches in the literature assume that a single relation r over relational schema $R(a_1, \ldots, a_n)$, with a_i an attribute on domain $D_i, i = 1, \ldots, n$, contains all sensitive information that needs to be protected [7–9]. Note however that the techniques that will be described in the following can also work with other data models. The privacy requirements are instead modeled through *confidentiality constraints*. A confidentiality constraint c over a relational schema $R(a_1, \ldots, a_n)$ is a subset of attributes in R[1] ($c \subseteq R$) meaning that for each tuple in r, the (joint) visibility of the values of the attributes in c is considered sensitive and must be protected. While simple, the definition of confidentiality constraints captures different protection requirements. In particular, depending on the attributes involved, confidentiality constraints can be classified in the following two categories.

- *Singleton constraints*. A singleton constraint states that the *values* assumed by the attribute in the constraint are considered sensitive and cannot be released (e.g., the `SSN` of the patients in a hospital is considered sensitive).

[1] When clear from the context R is used to denote either the relation schema R or the set of attributes in R.

Patient

SSN	Name	DoB	ZIP	Job	Illness	Physician
123-45-6789	A. Perry	75/12/22	22030	Nurse	Pneumonia	H. Daily
987-65-4321	B. Pott	71/03/18	22045	Employee	Diabetes	I. Dale
246-89-1357	C. Powal	65/06/14	22021	Manager	Hypertension	J. Dooley
135-79-2468	D. Prately	51/09/30	22030	Cook	Flu	K. Davis
753-19-8642	E. Preston	42/08/06	22041	Nurse	Gastritis	L. Denis
864-29-7531	F. Pickett	82/10/07	22020	Nurse	Flu	M. Dicks
264-81-5773	G. Pyne	68/04/24	22045	Employee	Pneumonia	N. Doe

(a)

c_0={SSN}
c_1={Name, DoB}
c_2={Name, ZIP}
c_3={Name, Illness}
c_4={Name, Physician}
c_5={DoB, ZIP, Illness}
c_6={DoB, ZIP, Physician}
c_7={Job, Illness}
c_8={Job, Physician}

(b)

Fig. 1. An example of relation (a) and of well defined constraints over it (b)

- *Association constraints*. An association constraint states that the *association* among the values of the attributes in the constraint is considered sensitive and cannot be released (e.g., the association of the name of patients with their illnesses must be protected).

Since the satisfaction of a confidentiality constraint c_i implies the satisfaction of any constraint c_j such that $c_i \subseteq c_j$, a set $\mathcal{C} = \{c_1, \ldots, c_m\}$ of confidentiality constraints is supposed to be *well defined*, that is, $\forall c_i, c_j \in \mathcal{C}, i \neq j,\ c_i \not\subset c_j$.

Example 1. Figure 1 illustrates an example of relation (Patient) along with a set of well defined confidentiality constraints, modeling the following privacy requirements:

- the list of SSNs of patients is considered sensitive (c_0);
- the association of patients' names with any other information in the relation but the job is considered sensitive ($c_1, \ldots, c_4$);
- attributes DoB and ZIP can work as a quasi-identifier [10] and therefore can be exploited to infer the identity of patients; their associations with both Illness and Physician are then considered sensitive (c_5 and c_6);
- the association between Job and Illness and the association between Job and Physician are considered sensitive (c_7 and c_8).

Note also that the association of patients' Name and SSN is sensitive and should be protected. However, such a constraint is redundant, because SSN has been declared sensitive (c_0): protecting SSN as an individual attribute implies automatic protection of its associations with any other attribute.

Given a relation r over schema $R(a_1, \ldots, a_n)$ and a set $\mathcal{C}$ of confidentiality constraints over R, the goal is to outsource the content of r in such a way that the sensitive associations represented as confidentiality constraints are protected. The approaches proposed in the literature for addressing such a problem are typically based on a possible combination between *fragmentation* and *encoding* techniques. Fragmentation consists in partitioning the attributes in R in different subsets (fragments), which are then outsourced in place of R. Formally, a fragment F_i of a relation R is defined as a subset of the attributes in R ($F_i \subseteq R$), while a *fragmentation* $\mathcal{F}$ is a set of fragments over R (i.e.,

$\mathcal{F} = \{F_1, \ldots, F_m\}$). The set of tuples of relation r over R projected on the attributes in F_i is a *fragment instance* over F_i. Intuitively, fragmentation protects sensitive associations by breaking them. Encoding means that the values of some attributes are obfuscated to make them readable only by authorized users. Although different encoding techniques can be adopted [7], we only consider *encryption* as an encoding technique. The approaches in the literature then differs in how the original relational schema R is fragmented to avoid the joint visibility of attributes involved in constraints and in how and whether encryption is used. In particular, existing approaches for enforcing confidentiality constraints can be partitioned into three different categories:

- *non-communicating pair of servers*, when R is partitioned into two fragments stored on two non-communicating servers and encryption (or another encoding technique) is used for protecting attributes when they cannot be stored in the two fragments without violating the constraints;
- *multiple fragments*, when R is partitioned into two or more disjoint fragments, possibly stored on the same server, and encryption is used within each fragment for maintaining in encrypted form all attributes not appearing in the clear;
- *departing from encryption*, when R is partitioned into two fragments, one stored at the data owner site and the other one stored at the external server. The two fragments can be joined by authorized users only and encryption is not used.

In the following, we present these three different strategies in more details. The discussion will focus on the different techniques that can be used to compute a fragmentation that satisfies the privacy requirements. The interested reader can refer to [11] for a detailed discussion on additional issues that arise in the data outsourcing scenario.

3 Non-Communicating Pair of Servers

The first proposal suggesting the use of fragmentation and encryption for outsourcing data while enforcing a set of confidentiality constraints has been presented in [7]. The basic idea consists in partitioning the original relational schema R into two fragments stored on two non-communicating servers, which do not know each other, thus preventing the joint visibility of attributes in the two fragments. We now describe the data fragmentation model and briefly illustrate how to compute a fragmentation.

3.1 Data fragmentation model

A relational schema R is partitioned into two fragments F_1 and F_2 stored at two non-communicating servers in such a way that the attributes involved in a confidentiality constraint cannot appear all together in a fragment. An encoding technique is used whenever an attribute cannot be stored within one of the two fragments without violating a confidentiality constraint. The encoding of an attribute $a \in R$ consists in representing the values of the attribute with two different attributes a^1 and a^2 included in the two fragments F_1 and F_2, respectively. The values of attribute a can then be reconstructed only by authorized parties opportunely combining the values of the two corresponding attributes a^1 and a^2. For instance, if attribute a is encrypted, then a^1 may contain the encrypted values of a and a^2 may contain the key(s) used for encrypting the values of attribute a. A fragmentation $\mathcal{F}$ is then defined as a triple $\langle F_1, F_2, E \rangle$, where E is the set of encoded attributes stored at both servers (i.e., $E \subseteq F_1$ and $E \subseteq F_2$) and $R = F_1 \cup F_2$. A fragmentation $\mathcal{F}$ is correct if $\forall c \in \mathcal{C}$ conditions $c \not\subseteq F_1$ and $c \not\subseteq F_2$ are both satisfied. At the physical level, a fragmentation $\mathcal{F} = \langle F_1, F_2, E \rangle$, with $F_1 = \{a_{1_1}, \ldots, a_{1_n}\}$, $F_2 = \{a_{2_1}, \ldots, a_{2_m}\}$, and $E = \{a_{e_1}, \ldots, a_{e_l}\}$ translates into two physical fragments with schema $F_1^e = \{\underline{\texttt{tid}}, a^1_{e_1}, \ldots, a^1_{e_l}, a_{1_1}, \ldots, a_{1_n}\}$ and $F_2^e = \{\underline{\texttt{tid}}, a^2_{e_1}, \ldots, a^2_{e_l}, a_{2_1}, \ldots, a_{2_m}\}$, respectively. Attribute `tid` is the primary key of both physical fragments and guarantees the *lossless join* property. The lossless join property guarantees that the content of the original relation r over R cal always be reconstructed through the join between the fragment instances over F_1^e and F_2^e. The join operation may be performed on the common attribute `tid` that can be either: *1)* the key attribute of R, if it is not sensitive, or *2)* an attribute that is added to both F_1^e and F_2^e during the fragmentation process, otherwise.

With this model, singleton constraints can only be satisfied by encoding the attributes in the constraints. Association constraints can instead be satisfied either by splitting the attributes in the constraints between F_1 and F_2, or by encoding at least one of the attributes in the constraints. Note however that it is not always possible to satisfy an association constraint via fragmentation. As a matter of fact, since there are only two fragments it may happen that the attributes involved in an association constraint cannot be split between the two fragments without violating another constraint. In these cases, it is necessary to apply an encoding technique on one of the attributes involved in the constraint.

Example 2. Consider relation Patient in Figure 1(a) and the set of well defined constraints over it in Figure 1(b). Suppose also that encryption is used as an encoding technique. Figure 2 illustrates the fragment instances over the physical fragments corresponding to the correct fragmentation $\mathcal{F} = \langle\{$`DoB`, `Illness`, `Physician`$\}$, $\{$`DoB`, `ZIP`, `Job`$\}$, $\{$`SSN`, `Name`$\}\rangle$. For simplicity, in this figure both encrypted values and corresponding keys are represented with Greek letters. Note that attribute `DoB` can be replicated without violating any constraint, thus improving query performance. Singleton constraint c_0 is enforced by encrypting attribute `SSN`. Association constraints $c_1, \ldots, c_4$ are satisfied by encrypting attribute `Name`. Finally, association constraints $c_5, \ldots, c_8$ are satisfied by fragmenting the involved attributes.

F_1^e

tid	SSN^1	Name^1	DoB	Illness	Physician
1	α	ϑ	75/12/22	Pneumonia	H. Daily
2	β	ι	71/03/18	Diabetes	I. Dale
3	γ	κ	65/06/14	Hypertension	J. Dooley
4	δ	λ	51/09/30	Flu	K. Davis
5	ε	μ	42/08/06	Gastritis	L. Denis
6	ζ	ν	82/10/07	Flu	M. Dicks
7	η	ξ	68/04/24	Pneumonia	N. Doe

F_2^e

tid	SSN^2	Name^1	DoB	ZIP	Job
1	o	τ	75/12/22	22030	Nurse
2	π	υ	71/03/18	22045	Employee
3	ϖ	ϕ	65/06/14	22021	Manager
4	ρ	φ	51/09/30	22030	Cook
5	ϱ	χ	42/08/06	22041	Nurse
6	σ	ψ	82/10/07	22020	Nurse
7	ς	ω	68/04/24	22045	Employee

Fig. 2. An example of a correct fragmentation in the non-communicating pair of servers scenario

3.2 Minimal fragmentation

Given a relational schema R and a set of well defined constraints $\mathcal{C}$ over it, there may exist different correct fragmentations. For instance, a fragmentation that encodes all attributes in R is always correct. However, such a fragmentation implies a higher query evaluation cost for authorized users than a fragmentation that minimizes the use of encoding and that resorts to fragmentation whenever possible. The query evaluation cost can be measured in different ways. In [7] the authors adopt an *affinity matrix*, which is a matrix with a row and a column for each attribute in R. Each entry $M[a_i, a_j]$, with $i \neq j$, represents the cost that would be paid in query execution if attributes a_i and a_j do not belong to the same fragment. Each entry $M[a_i, a_i]$, with $i = 1, \ldots, n$, represents the cost that would be paid if attribute a_i is encoded. The cost of a fragmentation $\mathcal{F}$ is then defined as the sum of the cells $M[a_i, a_j]$ in the matrix such that $a_i \in F_1$ and $a_j \in F_2$, and the cells $M[a_i, a_i]$ in the matrix such that $a_i \in E$.

The problem of computing a fragmentation with minimum cost is NP-hard since, as proved in [7], the hypergraph coloring problem [12] reduces to it. As a consequence, an algorithm that computes a fragmentation with minimum cost would operate in time exponential with the number of attributes in R. To avoid this inconvenience, in [7] the authors propose to combine known approximation algorithms used for solving the min-cut and the weighted set cover problems, obtaining three different heuristics working in polynomial time.

4 Multiple Fragments

The main problem of the approach illustrated in Section 3 is that it is based on the complete absence of communication among the storage servers (which have to be completely unaware of each other). This assumption is however difficult to enforce in practice and a collusion among the servers, or with an authorized user of the system, can breach the privacy of the data. The solution proposed in [8], and refined in [13, 14], removes the need of having two non-communicating pair of servers. Like for the previous solution, we first describe the data fragmentation model and then illustrate how to compute a fragmentation.

4.1 Data fragmentation model

The proposal illustrated in [8] uses fragmentation and encryption for enforcing a set $\mathcal{C}$ of confidentiality constraints defined over a relational schema R and produces a set of fragments. The resulting fragments can be stored on the same server since they cannot be joined for reconstructing the content of the original relation. A fragmentation $\mathcal{F}$=$\{F_1 \ldots F_n\}$ is therefore considered *correct* if the following conditions hold: *1)* $\forall c \in \mathcal{C}, \forall F \in \mathcal{F}$, $c \nsubseteq F$; *2)* $\forall F_i, F_j \in \mathcal{F}, i \neq j$, $F_i \cap F_j = \emptyset$. Condition 1 states that a single fragment cannot contain in the clear attributes that form a confidentiality constraint. Condition 2 states that the fragments are disjoint. At the physical level, a fragmentation $\mathcal{F}$=$\{F_1 \ldots F_n\}$, with F_i=$\{a_{i_1}, \ldots, a_{i_m}\}$, $i = 1, \ldots, n$, translates into a set of physical fragments F^e_i, $i = 1, \ldots, n$. Each physical fragment F^e_i contains all the attributes in F_i in the clear, while all the other attributes of R are encrypted. The reason for reporting all attributes of R (in either encrypted or clear form) in each of the physical fragment is to guarantee that any query can be executed by querying a single physical fragment. Formally, the schema of a physical fragment F^e_i corresponding to fragment F_i=$\{a_{i_1}, \ldots, a_{i_m}\}$ is F^e_i($\underline{salt}$,enc,$a_{i_1}, \ldots, a_{i_m}$) where:

- *salt* is the primary key of F^e_i and contains a randomly chosen value;
- *enc* contains the encryption of all the attributes of R that do not belong to the fragment (i.e. R - F_i), combined before encryption in a binary XOR ($\oplus$) with a salt;
- $a_{i_1}, \ldots, a_{i_n}$ correspond to the attributes in fragment F_i.

Note that to protect encrypted values from frequency-based attacks [15], a salt is applied on each encryption. Attribute *salt* of a physical fragment stores such values that due to their randomness can also be used as primary keys.

Singleton constraints can only be satisfied by encryption, that is, by preventing attributes in singleton constraints to appear in the clear within a fragment. Association constraints can be satisfied by storing the attributes composing the constraint in different fragments. This is always possible because if an attribute cannot be inserted in an existing fragment without violating a confidentiality constraint, then a new fragment can be created and the attribute can be inserted in it. In this way, we *maximizes the visibility* of the data since encryption is used only for protecting singleton constraints. A fragmentation $\mathcal{F}$ that satisfies all the confidentiality constraints and that maximizes data visibility is a fragmentation where each attribute that does not appear in singleton constraints belongs to *exactly* one fragment in $\mathcal{F}$. Clearly, a solution maximizing visibility permits a more efficient query evaluation.

Example 3. Consider relation PATIENT in Figure 1(a) and the set of constraints over it in Figure 1(b). An example of a correct fragmentation that maximizes visibility is $\mathcal{F}$=$\{$`{Name, Job}`, `{DoB, ZIP}`, `{Illness, Physician}`$\}$. Figure 3 illustrates the fragment instances over the physical fragments corresponding to $\mathcal{F}$. Note that only attribute `SSN` does not appear in the clear in the fragments since it belongs to a singleton constraint (c_0).

F_1^e

salt	enc	Name	Job
s_1^1	α	A. Perry	Nurse
s_2^1	β	B. Pott	Employee
s_3^1	γ	C. Powal	Manager
s_4^1	δ	D. Prately	Cook
s_5^1	ε	E. Preston	Nurse
s_6^1	ζ	F. Pickett	Nurse
s_7^1	η	G. Pyne	Employee

F_2^e

salt	enc	DoB	ZIP
s_1^2	ϑ	75/12/22	22030
s_2^2	ι	71/03/18	22045
s_3^2	κ	65/06/14	22021
s_4^2	λ	51/09/30	22030
s_5^2	μ	42/08/06	22041
s_6^2	ν	82/10/07	22020
s_7^2	ξ	68/04/24	22045

F_3^e

salt	enc	Illness	Physician
s_1^3	τ	Pneumonia	H. Daily
s_2^3	υ	Diabetes	I. Dale
s_3^3	ϕ	Hypertension	J. Dooley
s_4^3	φ	Flu	K. Davis
s_5^3	χ	Gastritis	L. Denis
s_6^3	ψ	Flu	M. Dicks
s_7^3	ω	Pneumonia	N. Doe

Fig. 3. An example of a correct fragmentation in the multiple fragments scenario

4.2 Minimal fragmentation

Given a relational schema R and a set $\mathcal{C}$ of well defined constraints over it, there may exist different correct fragmentations that maximize visibility. As an example, a fragmentation $\mathcal{F}$ composed of singleton fragments, one for each attribute that does not appear in a singleton constraint is a correct fragmentation. Such a fragmentation however makes query execution inefficient. As a matter of fact, queries defined on more than one attribute can be executed only with the involvement of the client. Like for the non communicating pair of servers scenario, it is important to identify, among all the correct fragmentations maximizing visibility, the one that minimizes the cost of query execution for the client. To this purpose, there are different metrics that can be adopted for measuring the quality of a fragmentation. A simple metric is the *number of fragments* composing a fragmentation $\mathcal{F}$ [8]. The rationale is that a low number of fragments implies that more attributes are stored in the clear in the same fragment, thus improving the efficiency in query execution. The problem of computing a fragmentation that minimizes the number of fragments is however NP-hard (the hypergraph coloring problem [12] reduces to this problem). To the aim of efficiently computing a correct fragmentation with a limited, even if not minimum, number of fragments, in [8] the authors introduce a definition of *minimality* that is based on the representation of a correct fragmentation that maximize visibility through a *fragment vector*. Given a fragmentation $\mathcal{F}$=$\{F_1,\ldots,F_m\}$ of a relational schema R, the fragment vector $V_{\mathcal{F}}$ representing $\mathcal{F}$ is a vector with an element $V_{\mathcal{F}}[a]$ for each attribute a in $\bigcup_{i=1}^{m} F_i$, where $V_{\mathcal{F}}[a]$ is set to F if attribute a belongs to fragment F.

Example 4. Consider fragmentation $\mathcal{F}$=$\{\{\texttt{Name, Job}\}, \{\texttt{DoB, ZIP}\}, \{\texttt{Illness, Physician}\}\}$ in Figure 3. The fragment vector representing $\mathcal{F}$ is defined as follows.

- $V_{\mathcal{F}}[\texttt{Name}] = V_{\mathcal{F}}[\texttt{Job}] = \{\texttt{Name}, \texttt{Job}\}$;
- $V_{\mathcal{F}}[\texttt{DoB}] = V_{\mathcal{F}}[\texttt{ZIP}] = \{\texttt{DoB}, \texttt{ZIP}\}$;
- $V_{\mathcal{F}}[\texttt{Illness}] = V_{\mathcal{F}}[\texttt{Physician}] = \{\texttt{Illness}, \texttt{Physician}\}$.

Fragment vectors define a partial order relationship, denoted $\prec$, among the correct fragmentations maximizing visibility of a relational schema R with respect to a set $\mathcal{C}$ of well defined constraints. In particular, a fragmentation $\mathcal{F}'$

dominates $\mathcal{F}$, denoted $\mathcal{F}\preceq\mathcal{F}'$, iff $V_{\mathcal{F}}[a]\subseteq V_{\mathcal{F}'}[a]$, for all attributes in R that do not belong to singleton constraints. Also, $\mathcal{F}\prec\mathcal{F}'$ iff $\mathcal{F}\preceq\mathcal{F}'$ and $\mathcal{F}\neq\mathcal{F}'$. In other words, a fragmentation $\mathcal{F}'$ dominates a fragmentation $\mathcal{F}$ if $\mathcal{F}'$ can be obtained by merging two (or more) fragments in $\mathcal{F}$.

Example 5. Consider relation Patient in Figure 1(a), the set of constraints over it in Figure 1(b), and the following two correct fragmentations that maximize visibility: $\mathcal{F}_1$={{`Name`, `Job`}, {`DoB`, `ZIP`}, {`Illness`, `Physician`}} and $\mathcal{F}_2$={{`Name`}, {`Job`}, {`DoB`, `ZIP`}, {`Illness`, `Physician`}}. Since $\mathcal{F}_1$ can be obtained by merging fragments {`Name`} and {`Job`} in $\mathcal{F}_2$, $\mathcal{F}_2\preceq\mathcal{F}_1$.

The problem of computing a fragmentation with a minimal number of fragments consists then in computing a fragmentation $\mathcal{F}$ that maximizes visibility and such that there is not a fragmentation $\mathcal{F}'$ maximizing visibility and correctly enforcing $\mathcal{C}$, such that $\mathcal{F}\prec\mathcal{F}'$. The algorithm proposed in [8] to solve this problem operates in $O(n^2 \cdot m)$, where n is the number of attributes in R, while m is the number of non singleton constraints in $\mathcal{C}$.

Alternative metrics that provide a more precise measure on the quality of a fragmentation are based on the use of an affinity matrix [13] or on the definition of a specific cost function that models the cost of evaluating a set of representative queries on $\mathcal{F}$ [14]. It is interesting to note that both the affinity matrix in [13] and the cost function in [14] are monotonic with respect to the dominance relationship $\preceq$. This means that the quality of a fragmentation increases with the increase of the the number of attributes represented in the clear in the fragmentation. In [14] the authors exploit this property and propose an exact algorithm for computing a fragmentation with minimum cost, which avoids to visit the whole space of solutions by exploiting relation $\preceq$ and the monotonicity of the cost function.

5 Departing from Encryption

A significant advantage of the solution based on multiple fragments is that it uses encryption only for protecting attributes involved in singleton constraints. However, the efficiency of query execution is still a problem since encryption causes a computational overhead for the client when executes a query, and for the data owner in key management. In [9, 16] the authors put forward the idea of completely departing from encryption. The proposed solution is based on the assumption that the data owner is willing to store a small portion of the data to guarantee the enforcement of confidentiality constraints.

5.1 Data fragmentation model

The proposal illustrated in [9] assumes that a subset of the data are stored at the data owner side, while the remaining information is outsourced to an external storage server. The input of the problem is still a relational schema R and a set $\mathcal{C}$ of confidentiality constraints defined over R. The result of the fragmentation process is a pair $\mathcal{F} = \langle F_o, F_s \rangle$ of fragments, where F_o is stored at the data owner side and F_s is stored at the storage server. A fragmentation $\mathcal{F}$ is considered *correct* if fragment F_s does not violate any constraint in $\mathcal{C}$ (i.e., $\forall c \in \mathcal{C}$ condition $c \not\subseteq F_s$ is satisfied) and all attributes of R appear in at least one fragment to avoid loss of information. Note that fragment F_o could possibly violate constraints, since it is stored at the data owner side that is supposed to be trusted and accessible only by authorized users. The solution in [9] also assumes that even if the data owner is willing to store a portion of the data, her storage capacity is limited. A first consequence of this assumption is that the information should not be replicated in the two fragments F_o and F_s. In other words, fragments F_o and F_s should be disjoint to avoid replication of attributes already stored at the server side also at the data owner side. The only attributes that the two fragments have in common is an identifier that is needed to guarantee the lossless join property. Such an identifier can correspond to the primary key of R, if it is not sensitive, or an attribute that does not belong to the relational schema R and that is added to both the fragments during the fragmentation process, otherwise. At the physical level, a fragmentation $\mathcal{F} = \langle F_o, F_s \rangle$, with $F_o = \{a_{o_1}, \ldots a_{o_i}\}$ and $F_s = \{a_{s_1}, \ldots a_{s_j}\}$ translates into two physical fragments $F^e_o(\underline{\texttt{tid}}, a_{o_1}, \ldots a_{o_i})$ and $F^e_s(\underline{\texttt{tid}}, a_{s_1}, \ldots a_{s_j})$, respectively, where `tid` is the common identifier.

Since data are only partially outsourced and data are not encrypted, singleton constraints can only be satisfied by storing the involved attributes at the data owner side. Also, association constraints can be satisfied only by fragmentation, that is, by storing at least one of the attributes in the constraint at the data owner side. Note that, in this case, all the constraints can be enforced by fragmentation, even if $\mathcal{F}$ is composed of two fragments only since F_o is supposed to be stored at a trusted party.

Example 6. Consider relation PATIENT in Figure 1(a) and the set of well defined constraints over it in Figure 1(b). An example of a correct fragmentation is F_o={`SSN`, `Name`, `ZIP`, `Job`} and F_s={`DoB`, `Illness`, `Physician`}. Figure 4 illustrates the fragment instances over the physical fragments corresponding to F_o and F_s. Constraint c_0 is satisfied by storing attribute `SSN` in F_o. Constraints $c_1, \ldots, c_4$ are satisfied by storing attribute `Name` in F_o. Constraints c_5 and c_6 are satisfied by storing attribute `ZIP` in F_o. Constraints c_7 and c_8 are satisfied by storing attribute `Job` in F_o.

5.2 Minimal fragmentation

Similarly to previous approaches, given a relational schema R and a set of well defined constraints $\mathcal{C}$ over it, there may exist different fragmentations that are correct and non-redundant. For instance, a fragmentation where F_o=R is obviously correct but it coincides with no outsourcing. Among all possible correct fragmentations, it is necessary to compute a solution that reduces either the storage at the data owner side, or the data owner's intervention in the query evaluation process (or both of them).

F_o^e

tid	SSN	Name	ZIP	Job
1	123-45-6789	A. Perry	22030	Nurse
2	987-65-4321	B. Pott	22045	Employee
3	246-89-1357	C. Powal	22021	Manager
4	135-79-2468	D. Prately	22030	Cook
5	753-19-8642	E. Preston	22041	Nurse
6	864-29-7531	F. Pickett	22020	Nurse
7	264-81-5773	G. Pyne	22045	Employee

F_s^e

tid	DoB	Illness	Physician
1	75/12/22	Pneumonia	H. Daily
2	71/03/18	Diabetes	I. Dale
3	65/06/14	Hypertension	J. Dooley
4	51/09/30	Flu	K. Davis
5	42/08/06	Gastritis	L. Denis
6	82/10/07	Flu	M. Dicks
7	68/04/24	Pneumonia	N. Doe

Fig. 4. An example of a correct fragmentation in the departing from encryption scenario

For computing a fragmentation that minimizes storage and computational burden for the data owner it is necessary to define a metric able to measure the cost of a fragmentation. In [9] the authors propose different metrics that depend on the resource whose consumption should be minimized (e.g., the storage space, the bandwidth capacity, the computational power) and on the information available at fragmentation time. For instance, a simple metric corresponds to the number of attributes in F_o. The minimization of the number of attributes implies a minimization of the storage space used at the data owner side as well as a minimization of the number of queries that require an involvement of the data owner. If, for example, is also available the information about the size of the attributes in R, then another possible metric consists in the total size of the attributes stored at the data owner side. More sophisticated metrics can be defined when information on the possible query workload is known.

In [9] the authors show that independently from the metric adopted to measure the cost of a fragmentation the problem of computing a fragmentation with minimum cost is NP-hard (the minimum hitting set problem [12] reduces to it in polynomial time). Therefore, in [9] the authors propose a heuristic algorithm that solves the problem in polynomial time with respect to the number of attributes in R. The main advantage of this algorithm is its flexibility, since it can be adopted with any metric.

6 Open Issues

The problem of satisfying confidentiality constraints in data outsourcing is becoming of great interest and different solutions have been proposed to the aim of maximizing the advantages of outsourcing, while preventing unauthorized accesses to sensitive information. There are however different issues that require further investigations and that we now briefly describe.

- *Multiple relations.* Most of the solutions proposed in the literature for privacy protection are based on the assumption that the sensitive information is stored in a single relation. An interesting direction that needs to be explored consists in assuming that data are represented through a set of relations that can be possibly joined.

- *Definition of confidentiality constraints.* The proposals illustrated in this paper are based on the assumption that confidentiality constraints are defined by the data owner according to her knowledge of the domain. However, the definition of a correct and complete set of confidentiality constraints is a critical and difficult task since it is necessary to consider the relationships among data. In particular, functional dependencies must be taken into account, since otherwise they could be exploited for inference attacks.
- *Data utility.* When publishing data, there are two contrasting needs that have to be taken into consideration: privacy protection and data utility. The fragmentation-based techniques described in this paper mainly focus on privacy protection and do not consider data utility. It would then be interesting to extend such proposals by exploring novel solutions that will take into consideration not only the privacy requirements but also explicit requests for views over data. These view requirements can be expressed, for example, as associations that have to be preserved during the fragmentation process.
- *Obfuscated associations.* Whenever the association among a set of attributes is considered sensitive (and it is therefore modeled as a confidentiality constraint), it is not possible to publish the involved attributes in a fragment, even if the utility of the data would considerably increase. It would be then interesting to define a solution that allows the publication of a sanitized/obfuscated version of sensitive but useful associations. The publication method should carefully handle the tradeoff between data utility, on one side, and association confidentiality, on the other side.
- *Metrics.* The computation of a minimum fragmentation implies the definition of a metric that is used to evaluate the cost of a fragmentation. A metric needs to take into consideration different parameters, such as, the storage at the data owner side, the computational resources required to the client for query evaluation, and the bandwidth occupation necessary for interactions among parties. Also, the metric adopted should be based on information that should be available to the data owner in advance with respect to the fragmentation process and that should be easy to compute. It would then be interesting to define sophisticate metrics able to capture the different parameters that may have an impact on the cost of a fragmentation.
- *Write operations.* A common aspect of all the proposals discussed is that they only support read operations. There are however different contexts where the consideration of read operations only may be a limitation (e.g., within a multi-owner context). It would then be interesting to extend current approaches for supporting write operations.

7 Conclusions

Fragmentation has been recently investigated as a technique for guaranteeing the protection of outsourced data. In this paper, we described three different solutions presented in the literature that possibly combine fragmentation and encryption and that produce a fragmentation correct with respect to the given privacy requirements. We then concluded the paper with a discussion on some open research challenges.

8 Acknowledgments

The paper is based on joint work with Valentina Ciriani, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. The authors would like to thank Pierangela Samarati for her comments, suggestions, and discussions that helped improving the organization of the paper. This work was supported in part by the EU within the 7FP project "PrimeLife" under grant agreement 216483.

References

1. Damiani, E., De Capitani di Vimercati, S., Jajodia, S., Paraboschi, S., Samarati, P.: Balancing confidentiality and efficiency in untrusted relational DBMSs. In: Proc. of the 10th ACM Conference on Computer and Communications Security (CCS 2003), Washington, DC, USA (October 2003)
2. Ceselli, A., Damiani, E., De Capitani di Vimercati, S., Jajodia, S., Paraboschi, S., Samarati, P.: Modeling and assessing inference exposure in encrypted databases. ACM Transactions on Information and System Security (TISSEC) **8**(1) (February 2005) 119–152
3. Hacigümüş, H., Iyer, B., Mehrotra, S., Li, C.: Executing SQL over encrypted data in the database-service-provider model. In: Proc. of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2002), Madison, WI, USA (June 2002)
4. Hacigümüş, H., Iyer, B., Mehrotra, S.: Providing database as a service. In: Proc. of 18th International Conference on Data Engineering (ICDE 2002), San Jose, California, USA (February 2002)
5. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Over-encryption: Management of access control evolution on outsourced data. In: Proc. of the 33rd International Conference on Very Large Data Bases (VLDB 2007), Vienna, Austria (September 2007)
6. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Encryption policies for regulating access to outsourced data. ACM Transactions on Database Systems (2010) (to appear).
7. 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: Proc. of the Second Biennial Conference on Innovative Data Systems Research (CIDR 2005), Asilomar, CA, USA (January 2005)
8. 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: Proc. of the 12th European Symposium On Research In Computer Security (ESORICS 2007), Dresden, Germany (September 2007)
9. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Keep a few: Outsourcing data while maintaining confidentiality. In: Proc. of the 14th European Symposium On Research In Computer Security (ESORICS 2009), Saint Malo, France (September 2009)

10. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering **13**(6) (November 2001) 1010–1027
11. Samarati, P., De Capitani di Vimercati, S.: Data protection in outsourcing scenarios: Issues and directions. In: Proc. of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2010), Beijing, China (April 2010)
12. Garey, M., Johnson, D.: Computers and Intractability; a Guide to the Theory of *NP*-Completeness. W.H. Freeman (1979)
13. 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 (2010) to appear.
14. 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: Proc. of the 29th International Conference on Distributed Computing Systems (ICDCS 2009), Montreal, Quebec, Canada (June 2009)
15. Schneier, B.: Applied Cryptography, 2/E. John Wiley & Sons (1996)
16. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Enforcing confidentiality constraints on sensitive databases with lightweight trusted clients. In: Proc. of the 23rd IFIP TC-11 WG 11.3 Seventeenth Annual Working Conference on Data and Application Security (DBSec 2009), Montreal, Quebec, Canada (July 2009)