Secure Data Outsourcing with Adversarial Data Dependency Constraints

Boxiang Dong
Stevens Institute of Technology
Hoboken, NJ, USA 07030
Email: bdong@stevens.edu

Wendy Wang
Stevens Institute of Technology
Hoboken, NJ, USA 07030
Email: hwang4@stevens.edu

Jie Yang
South China University of Technology
Guangzhou, China 510641
Email: yjclear@scut.edu.cn

***Abstract*—Cloud computing enables end-users to outsource their dataset and data management needs to a third-party service provider. One of the major security concerns of the outsourcing paradigm is how to protect sensitive information in the outsourced dataset. In general, the sensitive information can be protected by encryption. However, data dependency constraints in the outsourced data may serve as adversary knowledge and bring security vulnerabilities. In this paper, we focus on functional dependency (FD), an important type of data dependency constraints, and study the security threats by the adversarial FDs. We design the practical scheme that can defend against the FD attack by encrypting a small amount of non-sensitive data (encryption overhead). We prove that searching for the scheme that leads to the optimal encryption overhead is NP-complete, and design efficient heuristic algorithms. We conduct an extensive set of experiments on two real-world datasets. The experiment results show that our heuristic approach brings small amounts of encryption overhead (at most 1% more than the optimal overhead), and enjoys a ten times speedup compared with the optimal solution. Besides, our approach can reduce up to 90% of the encryption overhead of the-state-of-art solution.**

1. Introduction

Recent years have witnessed increasing need for large amounts of digital information to be collected and studied. To address the big data need, the database-as-a-service (DAS) model was introduced [8], facilitated by the evolution of cloud computing [12]. It allows end-users with limited resources to outsource their private datasets to a third-party service provider [9]. Since the service provider may not be fully trusted, the DAS model raises a few security issues. One of the issues is how to protect the sensitive information in the outsourced data. A general solution is to encrypt the data so that the service provider cannot access the original data without a proper decryption key.

However, recent study has shown that deterministic encryption at a fine granularity may not be robust against adversary data dependency constraints [3]. A typical type of data dependency constraints is the *functional dependency* (FD). Traditionally, FDs are statements of value constraints between attributes in a relation. Informally, a $FD: X \rightarrow Y$ constraint indicates that attribute set X uniquely determines attribute set Y. For instance, the FD *Zipcode→City* indicates that all tuples of the same zipcode values always have the same *City* values. FDs play an important role in relational theory and relational database design. They have been thoroughly researched and applied to improving schema quality through normalization [1], [4], [10] and to improving data quality in data cleaning [6], [2].

NM	SEX	AGE	DC	DS
Alice	F	53	CPD5	HIV
Carol	F	30	VPI8	Breast Cancer
Ela	F	24	VPI8	Breast Cancer

(a) The original dataset D

NM	SEX	AGE	DC	DS
Alice	F	53	CPD5	α
Carol	F	30	VPI8	Breast Cancer
Ela	F	24	VPI8	γ

(b) The unsafe encrypted dataset $\bar{D}$

Figure 1. An example of data instance (FD: DC $\rightarrow$ DS)

It is possible that FDs can serve as an important piece of adversary knowledge and bring security vulnerabilities. For example, consider a hospital that stores its patients' information, including patient's name (NM), gender (SEX), age (AGE), hospital-wide disease code (DC) and disease (DS), in a dataset. An example of data instance is shown in Figure 1 (a). Assume it has the FD: F: DC $\rightarrow$ DS. Consider the following patients' security settings: Alice and Ela require that their disease information cannot be shared with any third-party, while Carol agrees to share her disease information with a third-party. Before outsourcing the dataset to a third-party service provider for data management and analysis, the hospital encrypts its data according to the patients' settings. The encrypted instance is shown in Figure 1 (b). Since Ela's and Carol's records have the same *DC* value, the attacker can easily infer Ela's disease from Carol's record if he has the knowledge of F.

In practice, adversarial FD constraints are easily accessible from common sense or other data sources. Therefore, it is vital to design robust approaches that can defend against the attacks based on adversarial FDs. A straightforward approach is to encrypt all data values in the dataset (possibly at the finest granularity) regardless whether they are involved in any FD. However, as the data owner may only consider a portion of her dataset as sensitive, encrypting all data values

may over-protect the data and dramatically reduce the data usability. Only encrypting a (possibly small) portion of *non-sensitive* data values besides the sensitive values is sufficient to defend against the FD attack. Our goal is to find the optimal scheme that encrypts the minimal amounts of non-sensitive data while providing strong guarantee against the attacks based on adversarial FDs.

In this paper, we make the following contributions. First, we define a flexible security model that allows the client to express content-based security granularity in the format of *security constraints* (SCs). The SCs, which are expressed in selection-projection SQL queries, can support an extensive set of security settings. Second, we perform the foundational study of security vulnerabilities by adversarial FDs, and formalize the *FD attack*. We also investigate under which circumstances the FD attack does not bring any security leakage. Third, we show that the existing sufficient conditions against the FD attack in the state-of-art solution are not complete, and investigate the complete sufficient conditions against the FD attack. Following this, we design the encryption approaches that find the data cells that are necessarily to be encrypted to defend against the FD attack. We prove that finding the optimal scheme that encrypts the minimum number of data cells is NP-complete. Therefore, we design efficient greedy algorithms to find a small amount of non-sensitive data cells to be encrypted (together with sensitive data cells), with the presence of one or multiple SCs. Last but not least, we launch an extensive set of experiments on real datasets to evaluate the performance of our approaches. The experimental results demonstrate that our greedy approach is ten times faster than the optimal solution, with as small as 1% of the overhead compared with the state-of-art solution [16].

The rest of the paper is organized as follows. Section 2 describes the preliminaries. Section 3 introduces the FD attack and various schemes that can defend against the attack. Section 4 presents our approach to defend against FD attack. Section 5 provides the experiment results. Section 6 discusses the related work. Section 7 concludes the paper.

2. Preliminaries

In this section, we introduce the preliminaries.

Outsourcing Framework. We consider the framework that contains three parties: the data owner, the users (clients), and the service provider (server). The data owner has a dataset D that will be outsourced to the server. In this paper, we consider D as a relational table that consists of m attributes and n records. We use $r[X]$ to specify the value of record r on the attribute(s) X. Since D may contain sensitive information and the server is not fully trusted, the data owner encrypts D to $\bar{D}$ using symmetric encryption algorithms, and sends $\bar{D}$ to the server. Only authorized users have the key to access the private data. It is worth noting that it is possible that only a portion of D is sensitive. The non-sensitive data of D is allowed to be accessible by the server.

In this paper, we consider deterministic encryption mechanism. The algorithm is applied at cell level, i.e., the cells in the relation D are encrypted individually.

Functional Dependency. We assume the dataset D has data dependency constraints that are in the format of *functional dependency*. Formally, given a relational dataset D, a set of attributes Y of D is said to be *functionally dependent* on another set of attributes X of D (denoted $X \rightarrow Y$) if and only if for any pair of records r_1, r_2 in D, if $r_1[X] = r_2[X]$, then $r_1[Y] = r_2[Y]$. We use $LHS(F)$ and $RHS(F)$ to denote the attributes at the left-hand (right-hand, resp.) side of the functional dependency F. A FD $F : X \rightarrow Y$ is said as *trivial* if $Y \subseteq X$. In this paper, we only consider non-trivial FDs. It has been well known that any FD F whose $RHS(F)$ contains more than one attribute can be decomposed into multiple FD rules, each with a single attribute of $RHS(F)$ at its right side. Therefore, for the following discussions, we only consider FD rules that contain one single attribute at the right hand side. We assume that the client is aware of the FDs in her dataset before outsourcing.

Security Constraints. We propose *security constraints* (SCs) as a means for the data owner to specify the data that she intends to protect from the untrusted service provider. We adapt the *selection-projection queries* [3] as the format of the SCs. Formally, the *security constraint* is defined in the format $\Pi_Y \sigma_c$, where Π_Y denotes the projection of a relation on attributes Y, and σ_c denotes the selection condition in which C is a conjunction of equalities of format $A = B$ or $A = a$ such that A, B are attributes and a is a constant. Using selection-projection queries enables SCs to be expressed in SQL statements so that the data owner can easily enforce the SCs (by executing the SQL statements) over her dataset. For a given dataset D and a SC S, we use $S(D)$ to denote the *sensitive* data that is required to be protected. We consider encryption as the way to enforce SCs. We define the *basic scheme* as following.

Definition 2.1. **[Basic Scheme]** *Given a dataset D and a set of security constraints $\mathcal{S}$, the* basic scheme *of enforcing $\mathcal{S}$ on D is to encrypt $\cup_{S \in \mathcal{S}}(S(D))$.*

Example 2 has shown a table instance D and its SC (Figure 2 (a)), as well as the basic encryption scheme (Figure 2 (b)) of D according to the given SC.

For any SC $S : \Pi_Y \sigma_c$, we use $Proj(S)$ to denote the projection attribute list Y of Π_Y, and $Sel(S)$ to denote the list of the attributes in its selection condition σ_c. We allow $Sel(S)$ to be empty. For example, consider the security constraint S: $\Pi_B \sigma_{C=c_1}$, then $Proj(S) = \{B\}$, and $Sel(S) = \{C\}$.

3. FD Attack

The basic idea of the FD attack is to find the *sensitive* and *evidence* records. We first define the sensitive/evidence records and sensitive cells.

Definition 3.1. **[Sensitive Cells/records, Evidence Records]** *Given the dataset D and a set of security constraints $\mathcal{S}$, let $\bar{D}$ be the basic scheme of enforcing $\mathcal{S}$ on D. Let $F : X \rightarrow Y$ be a FD of D. Then for each record $r \in \bar{D}$, if $r[Y]$ is encrypted, we call r a* sensitive record*, and $r[Z]$ a* sensitive cell*, for any Z that is a single attribute of Y. Given a sensitive record r, for*

TID	A	B	C
r_1	a_1	b_1	c_1
r_2	a_1	b_1	c_2
r_3	a_1	b_1	c_3
r_4	a_2	b_2	c_3

(a) Original table D ($FD: A \rightarrow B$, SC: $\Pi_B \sigma_{C=c_1}$)

TID	A	B	C
r_1	a_1	$\boldsymbol{\beta_1}$	c_1
r_2	a_1	b_1	c_2
r_3	a_1	b_1	c_3
r_4	a_2	b_2	c_3

(b) Basic encryption $\bar{D}$ (not robust)

TID	A	B	C
r_1	$\boxed{\boldsymbol{\alpha_1}}$	$\boldsymbol{\beta_1}$	c_1
r_2	a_1	b_1	c_2
r_3	a_1	b_1	c_3
r_4	a_2	b_2	c_3

(c) Solution 1 (overhead: 1)

TID	A	B	C
r_1	a_1	$\boldsymbol{\beta_1}$	c_1
r_2	a_1	$\boxed{\boldsymbol{\beta_1}}$	c_2
r_3	$\boxed{\boldsymbol{\alpha_1}}$	b_1	c_3
r_4	a_2	b_2	c_3

(d) Solution 2 (overhead: 2)

Figure 2. An example of FD attack and two fix approaches (encrypted cells are in bold; encrypted non-sensitive cells are in boxes)

each record $r' \neq r$ that none of $r'[X]$ and $r'[Y]$ in $\bar{D}$ is encrypted, but $r'[X] = r[X]$, we call r' an evidence record *of $r[Y]$.*

As an example, consider the table D and its basic scheme $\bar{D}$ in Figure 2 (a) and (b). The record r_1 is a sensitive record, with $r_1[B]$ as a sensitive cell. The records r_2 and r_3 are the evidence records of $r_1[B]$. It is worth noting that a sensitive record can contain both sensitive and non-sensitive cells. Take r_1 as an example. It contains the sensitive cell $r_1[B]$, and non-sensitive cells $r_1[A]$ and $r_1[C]$.

Given an FD $F: X \rightarrow Y$, consider a record $r \in \bar{D}$ such that $r[Y]$ is encrypted but $r[X]$ is not. The attacker can launch the FD attack to break the encryption on $r[Y]$. In particular, for any sensitive (and encrypted) value $r[Y] \in \bar{D}$, the FD attack on $r[Y]$ consists of the following steps:

Step 1: Search for the record $r' \in \bar{D}$ such that: (1) both $r'[X]$ and $r'[Y]$ are not encrypted, and (2) $r'[X] = r[X]$. Obviously these records are the evidence of $r[Y]$;

Step 2: If any evidence record exists, replace the (encrypted) $r[Y]$ with $r'[Y]$.

As an example, consider the base table D in Figure 2 (a) that has the FD $F: A \rightarrow B$. Also consider the security constraint $\Pi_B \sigma_{C=C_1}$ and the basic scheme $\bar{D}$ shown in Figure 2 (b). The attacker can easily find tuple r_2 as the evidence of $r_1[B]$, and replaces $r_1[B]$ in $\bar{D}$ (Figure 2 (b)) to be b_1 accordingly. We say a record r is *FD-compromised* if its encrypted value can be replaced with a plaintext value via the FD attack. We say an encrypted dataset $\bar{D}$ is *robust* against the FD attack if no record in $\bar{D}$ is FD-compromised. Otherwise $\bar{D}$ is *unsafe* against the FD attack.

4. Defending Against FD Attack

In this section, we discuss how to design robust schemes against the FD attack.

4.1. Robustness Checking of Basic Encryption

The basic scheme encrypts all sensitive values according to the given security constraints. First, we discuss how to check whether the basic scheme is robust against the FD attack. A naive method is to traverse the dataset finding whether there exist an encrypted cell in the basic scheme that has any evidence record. The complexity is $O(tn)$, where t is the number of sensitive cells and n is the size of D. Though correct, this approach can be costly for large datasets. Therefore, we design a method that can decide whether the basic scheme is robust against the FD attack by using FDs and SCs only, without traversing the dataset.

Our method relies on the sufficient condition of FD attack. [16] presents a *sufficient* condition for preventing the security leakage by FD inference as to ensure that for any FD $F: X \rightarrow Y$, the security level of X should be no lower than that of Y . We argue that this is not a complete set of sufficient conditions. We have the following lemma to show our complete sufficient conditions.

Lemma 4.1. *Consider a dataset D and its FD $F: X \rightarrow Y$. An scheme $\bar{D}$ is robust against the FD attack if for each record in $\bar{D}$ such that $r[Y]$ is sensitive, it satisfies one of the two conditions below:*
(Condition 1.) There exists at least one attribute $A \in X$ such that $r[A]$ is encrypted;
(Condition 2.) $r[X]$ is plaintext, but there does not exist a record $r' \neq r \in \bar{D}$ such that both $r'[X]$ and $r'[Y]$ are plaintext and $r'[X] = r[X]$.

Condition 1 of Lemma 4.1 is equivalent to the sufficient condition of [16], requiring that both $r[X]$ and $r[Y]$ must be encrypted at the same time. It is not necessary that all attributes of $r[X]$ must be encrypted; at least one attribute being encrypted is sufficient. Condition 2 addresses the case that $r[X]$ can remain to be plaintext as long as there is no evidence tuple of $r[Y]$ in $\bar{D}$. This condition is not covered by [16].

Following Lemma 4.1, we design the method of checking the robustness of the basic encryption scheme against the FD attack. First, we consider the case that there is a single security constraint (SC). We have the following theorem to check if the basic scheme is robust against the FD attack when one single SC is present.

Theorem 4.1. *Given a dataset D, a security constraint S, and an FD $F: X \rightarrow Y$, the basic scheme of D is robust against the FD attack w.r.t. F if one of the following conditions is met: (a) $X \cap Proj(S) \neq \emptyset$; (b) $Y \not\subseteq Proj(S)$; and (c) $Sel(S) \subseteq X \cup Y$.*

For example, consider the base table in Figure 2 (a), with FD $A \rightarrow B$ and SC $\Pi_B \sigma_{C=c_1}$. The basic scheme (Figure 2 (b)) does not satisfy any of the three conditions in Theorem 4.1. Therefore it is not robust against the FD attack.

The reasoning of robustness checking for one single SC can be easily extended to multiple SCs. We have the following theorem.

Theorem 4.2. *Given a dataset D, a set of security constraints $\mathcal{S}$, and an FD $F: X \rightarrow Y$ of D, the basic scheme of D is robust against the FD attack w.r.t. F if one of the following conditions is met: (1) $\forall S \in \mathcal{S}$, $X \cap Proj(S) \neq \emptyset$; (2) $Y \not\subseteq \cup_{S \in \mathcal{S}} Proj(S)$; (3) $\cup_{S \in \mathcal{S}} Sel(S) \subseteq X \cup Y$.*

4.2. Encryption Against FD attack

If the basic scheme is not robust against the FD attack, we design the method to fix the basic encryption. The key

idea of our fix method is to encrypt a set of non-sensitive cells additionally. In this section, we explain how to find those non-sensitive cells.

One Single SC. The success of the FD attack is the existence of evidence records. Therefore, we aim to find the evidence records and fix the encryption on them. First, we explain how to find the sensitive/evidence records efficiently. For a given security constraint $S : \Pi_Y \sigma_c$, in which Y is the RHS of the given FD F, first, we re-write S to be $S' : \Pi_{(X,Y)}\sigma_c$, where X and Y are LHS and RHS of F. Second, we apply S' on D, and bucketize $S'(D)$ by its values. All (X, Y) pairs of the same values are put into the same bucket. Each bucket represents a unique sensitive cell and its associated X value. Third, for each bucket $B(X = x, Y = y)$, we construct a selection-projection query $S_E : \Pi_{(X,Y)}\sigma_{(X=x,Y=y)}$, and apply S_E on D. Apparently, the bucket $B(X = x, Y = y)$ contains sensitive records of $Y = y$ with $X = x$, and $S_E(D)$ contains sensitive and evidence records of $Y = y$ where $X = x$.

Let n_S be the number of records in the bucket $B(X = x, Y = y)$, and n_E be the size of $S_E(D)$. Apparently if $n_S = n_E$, then there is no evidence record of the sensitive cell $Y = y$ that are associated with $X = x$ (i.e., it is robust against the FD attack). Otherwise, the basic scheme has to be fixed against the FD attack. Next, we describe two fix approaches to construct the schemes that achieve the sufficient conditions against the FD attack.

- Scheme 1: for each record r in the bucket $B(X = x, Y = y)$, randomly pick one attribute $A \in X$ and encrypt $r[A]$. It is possible that different A is picked for different records.
- Scheme 2: for each record r' in $S_E(D) - B(X = x, Y = y)$ (i.e., r' in $S_E(D)$ but not in $B(X = x, y = y)$), encrypt $r'[X]$ or $r'[Y]$.

We have the following theorem.

Theorem 4.3. *Consider a dataset D, an FD $F : X \to Y$, and a SC S, applying one of the two schemes above always delivers a robust scheme against the FD attack.*

Both fix schemes require to encrypt some *non-sensitive* cells. We measure the number of those non-sensitive cells as the *encryption overhead* against the FD attack. Formally,

Definition 4.1. **[Encryption Overhead]** *Given a dataset D and a set of SCs $\mathcal{S}$, let $\bar{D}$ be the basic scheme, and $\hat{D}$ be the dataset that is robust against the FD attack. Let e and e' be the number of encrypted cells in $\bar{D}$ and $\hat{D}$. Then the* encryption overhead *against the FD attack on D is $o = e' - e$.*

Our goal is to find an scheme that is robust against the FD attack with minimal encryption overhead. Apparently, for each sensitive cell, the encryption overhead of Scheme 1 is n_S, while the overhead of Scheme 2 is $n_E - n_S$. We pick the scheme whose overhead equals

$$min_o = min(n_S, n_E - n_S) \quad (1)$$

to decide the scheme of less encryption overhead. We do this for each sensitive cell. Due to the fact that the sensitive/evidence records of different sensitive cells do not overlap, the final scheme is guaranteed to return the minimal encryption overhead. The complexity is $O(n^2)$, where n is the size of D.

In the example in Figure 2, apparently, $\bar{D}$ is not robust against the FD attack. Figure 2 (c) & (d) show two solutions to fix $\bar{D}$ by our two fix schemes aforementioned. Scheme 1 (Figure 2 (c)) encrypts the a_1 value of the sensitive record r_1, with the encryption overhead 1. Scheme 2 (Figure 2 (d)) encrypts either the A or B attribute of the evidence records r_2 and r_3, with the encryption overhead 2. We pick Scheme 1 due to its smaller overhead.

Multiple SCs. When there are $k > 1$ SCs $\mathcal{S} = \{S_1, \ldots, S_k\}$, the brute-force solution is to enumerate all possible encryption solutions, and pick the one of the smallest encryption overhead. There are 2^{tk} schemes in total, where t is the average number of sensitive cells of each SC. As each encryption solution needs $O(n^2)$ complexity to locate the sensitive and evidence records, the complexity of the brute-force method is $O(2^{tk}n^2)$, where n is the size of the dataset. This could be prohibitive if k is large.

A seemly straightforward method is to pick *local optimal* scheme for each SC $S_i \in \mathcal{S}$. However, the local optimality cannot guarantee the global optimal solution for the k SCs in terms of encryption overhead. For example, consider a dataset D that has an FD $X \to Y$ and two SCs S_1 and S_2. Assume S_1 considers the Y attribute of records r_1 and r_2 as sensitive, while S_2 considers the Y attribute of records r_3, r_4, r_5 as sensitive. For S_1, the evidence records are r_3, r_4, and r_5, while for S_2, the evidence records are r_1, r_2, and r_3. The local optimal solution of S_1 is to encrypt the X attribute of $\{r_1, r_2\}$, while the local optimal solution of S_2 is to encrypt the X attribute of $\{r_4, r_5\}$. The total encryption overhead is 4. However, the global optimal solution is to encrypt the X attribute of $\{r_1, r_2, r_3\}$ or $\{r_3, r_4, r_5\}$. The encryption overhead is 3.

Next, we prove that the problem of finding the optimal scheme of the minimal encryption overhead with multiple SCs is NP-complete.

Theorem 4.4. *Given a dataset D and $k > 1$ SCs $\mathcal{S}$, the problem of finding the optimal robust scheme that enforces $\mathcal{S}$ on D against the FD attack is NP-complete.*

The proof is based on the reduction to the *weighted vertex cover* problem. We omit the detailed proof due to the space limit.

Given the NP-completeness result, we design an efficient greedy algorithm (*GMM*) to find an scheme to defend against the FD attack with small encryption overhead. The key idea of *GMM* algorithm is to repeatedly pick minimum number of sensitive/evidence records, until the picked records cover all SCs. In particular, given k SCs $\mathcal{S} = \{S_1, \ldots, S_k\}$ for which the basic scheme is not robust, for each $S_i \in \mathcal{S}$ $(1 \leq i \leq k)$, it is associated with a set of triples $H_i = \{(v_{i,j}, s_{i,j}, e_{i,j})\}$, where $v_{i,j}$ is the j-th sensitive cell of S_i, and $s_{i,j}$ ($e_{i,j}$, resp.) is a set of sensitive records (resp. evidence records) of $v_{i,j}$. We store the record IDs in $s_{i,j}$ and $e_{i,j}$. We use $|s_{i,j}|$ ($|e_{i,j}|$, resp.) to denote the number of records in $s_{i,j}$ ($e_{i,j}$, resp.). For a

given H_i, we define $min_c = min_{\forall v_{i,j} \in H_i} min(|s_{i,j}|, |e_{i,j}|)$. We use min_v to denote the sensitive cell of min_c, and min_{se} to denote the set of sensitive/evidence records of min_v that deliver min_c. For instance, consider $H_i = \{((x_1, y_1), \{r_1, r_2\}, \{r_3, r_4, r_5\}), ((x_2, y_2), \{r_3, r_4\}, \{r_5\})\}$. Then $min_c = 1$, $min_v = \{(x_2, y_2)\}$, and $min_{se} = \{r_5\}$. Let $\mathcal{H} = \{H_1, \ldots, H_k\}$. Initially we mark every H_i in $\mathcal{H}$ as *uncovered*. First, we select $H_i \in \mathcal{H}$ whose min_c is the smallest among all uncovered H_i $(1 \leq i \leq k)$. Let min_s and min_e be the set of sensitive records and evidence records of the picked min_v. Second, for all triples $(v_{i,j}, s_{i,j}, e_{i,j}) \in \mathcal{H}$, if $v'_{i,j} = min_v$, we update $s_{i,j} = s_{i,j} - min_{se}$, and $e_{i,j} = e_{i,j} - min_{se}$. If there exist any H_i such that for each triple in H_i, $s_{i,j}$ or $e_{i,j}$ becomes empty, we mark H_i as *covered*. We repeat these two steps until all H_is in $\mathcal{H}$ are covered. The complexity of the *GMM* algorithm is $O(ktn^2)$, where k is the number of SCs, t is the average number of sensitive cells, and n is the size of D.

5. Experiments

5.1. Setup

Experiment environment. We implement our approaches in Java. All the experiments were executed on a machine with 2.4GHz Intel Core i5 CPU and 4GB memory running Mac OS X 10.9. The time performance and encryption overhead are measured as the average of five runs.

Datasets. We execute our experiments on two real-world datasets named *Adult* dataset[1] and the *Orders* table from the TPC-H benchmark datasets. To measure the scalability of our approaches, we construct a set of datasets of various sizes, by using *Adult* and *Orders* datasets as the seed.

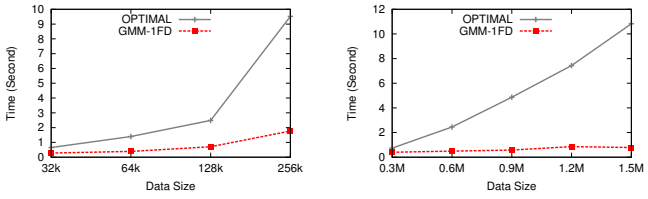
Approaches In the experiments, we implement the following two methods and evaluate them on the datasets.

- **GMM**: our heuristics approach (Section 4);
- **OPTIMAL**: the exhaustive search algorithm that finds the solution with the minimum encryption overhead.

We take *OPTIMAL* as the baseline methods and compare their performance with our approach.

5.2. Experiment Results

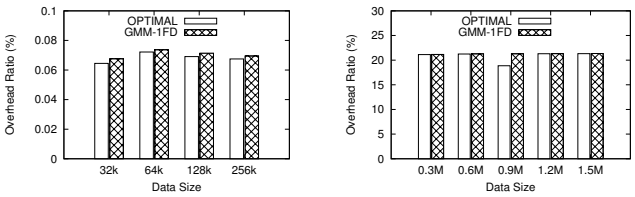
For this set of experiments, we randomly generate 10 SCs from the two datasets. We follow Theorem 4.1 to ensure that the generated SCs are potentially unsafe, meaning their basic scheme is not robust against the FD attack.



(a) Adult dataset (b) Orders dataset

Figure 3. Time Performance

1. Adult dataset is available at UCI Machine Learning Repository: http://archive.ics.uci.edu/ml/index.html.



(a) Adult dataset (b) Orders dataset

Figure 4. Encryption Overhead

Time Performance. We measure the time of our *GMM* algorithm on datasets of various sizes. We report the results in Figure 3. From the two figures, we observe that our *GMM* approach is much more efficient than the optimal scheme. In all circumstances, our *GMM* approach enjoys a ten times speedup compared with *OPTIMAL*. Another observation is that the time performance increases linearly with the data size. This is consistent with our expectation. One interesting observation is that even though the *Orders* datasets are much more larger than the *Adult* datasets, the time performance is comparable. This is mainly because the SCs of *Adult* datasets cover a large number of sensitive cells, while the SCs of *Orders* datasets cover a relatively smaller number of sensitive cells. Therefore, even though the *Adult* dataset is smaller, there is no substantial difference in the execution time on the two datasets.

Encryption Overhead. To measure the encryption overhead, we define *encryption overhead ratio* as $o = \frac{h}{n}$, where h is the encryption overhead, and n is the number of records of the dataset. We report the overhead of both our *GMM* approach and the *OPTIMAL* approach in Figure 4. First, we observe that our *GMM* approach delivers comparable overhead as *OPTIMAL*. In almost all the cases, the overhead brought by *GMM* is no greater than the optimal overhead by 1% (with only one exception in the *Orders* dataset of 0.9 million records). This encouraging result verifies that our *GMM* approach can efficiently find near-optimal encryption solution to defend against the FD attack. Second, the encryption overhead ratio is small, especially on the *Adult* dataset (no larger than 0.1%). This shows that our approach can be scaled to large datasets with small amounts of encryption overhead. Considering the state-of-the-art solution [16] that simply encrypts all data cells of the RHS of a FD, our *GMM* approach's encryption overhead is significantly smaller. Third, the encryption overhead ratio of the *Adult* dataset is much smaller than that of the *Orders* dataset. The reason why the *Adult* dataset has a small overhead ratio is that each sensitive cell is associated with only a small number (below 10) of sensitive and evidence records. On the contrary, even though the number of sensitive cells on the *Orders* dataset is smaller, the overhead ratio is larger because its sensitive cells are associated with a large number of sensitive and evidence records.

6. Related Work

In this section, we present previous work that is related to our study in this paper.

Encryption in DAS model. The database-as-a-service (DAS) was first introduced by Hacigumus et al. [8]. Its security issues have caught much attention in recent years.

To protect the sensitive data from the server itself, the client may encrypt her data before outsourcing. For example, searchable encryption [14] allows to conduct keyword searches on the encrypted data, without revealing any additional information. However, it cannot defend against the FD attack if the data is encrypted at a fine granularity. Homomorphic encryption [13] enables the service provider to perform meaningful computations on the encrypted data. It provides general privacy protection in theory, but it is not practical.

Inference attack in MDB. The inference channel problem via FDs has been considered in the context of multilevel secure relational database management system (MDB). The existing techniques are classified into two categories: (1) during database-design time, and (2) during query-time [3]. Most of the techniques that detect and remove inference channels during the database-design time modify the database design by increasing the classification levels of some data items (e.g. [7], [11], [15]). These techniques often result in over-classification of data [3]. Su et al. [16] investigate the inference problems due to functional dependencies in a multilevel Relational Database. They formally define *FD-compromises* under the attribute classification scheme as inference of unauthorized information based on the FD knowledge and the associated mapping. Besides proving that incurring minimum information loss to prevent FD-compromises is an NP-complete problem, they present an exact algorithm that leads to minimum information loss. However, this algorithm is only able to handle the security constraints at the attribute-level.

Regarding the query-time techniques, Brodsky et al. [3] present an integrated security mechanism which guarantees data confidentiality by extending a mandatory access control mechanism. They apply the chase process to generate all the information that could be inferred according to the answer to the user's past and current queries and the functional dependencies. If no security violation is detected, the query is answered. Otherwise, the query is rejected. Due to the fact that the client does not have the query evaluation capabilities, our work differs from [3] in that we enforce security via database encryption instead of controlling the query answers.

K-anonymity. An alternative to protect privacy of the outsourced dataset is to anonymize the data in a certain way. A data release is said to provide *k-anonymity* privacy protection if in the released data, each individual's information can not be distinguished from at least k-1 other individuals' [17]. Suppression and generalization are the two popular methods to achieve k-anonymity. Previous study [18] has shown that these two anonymization techniques may not be able to achieve k-anonymity with the existence of adversary's FD knowledge.

7. Conclusion

In this paper, we investigate the inference attacks based on adversary data dependency on the encrypted data in the DAS model. We focus on FDs as the adversary knowledge, and formalize the FD attack. We prove that finding an optimal scheme that is secure against the FD attack with the minimal encryption overhead is NP-complete. We design efficient heuristic approaches to construct robust schemes. For the future work, we are interested in extending the reasoning to the case when there exist multiple FDs. An another interesting extension is to consider association rules with high *support* and *confidence* [5], which introduce uncertainty to the inference attack.

8. Acknowledgements

This work is supported by the US National Science Foundation CAREER Grant #1350324 and the US National Science Foundation EAGER Grant #1464800.

References

[1] C. Batini, M. Lenzerini, and S. B. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys (CSUR)*, 1986.

[2] P. Bohannon, W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis. Conditional functional dependencies for data cleaning. In *ICDE*, 2007.

[3] A. Brodsky, C. Farkas, and S. Jajodia. Secure databases: Constraints, inference channels, and monitoring disclosures. *TKDE*, 2000.

[4] L. Chiticariu, M. A. Hernández, P. G. Kolaitis, and L. Popa. Semi-automatic schema integration in clio. In *VLDB*, 2007.

[5] G. Cormode, L. Golab, K. Flip, A. McGregor, D. Srivastava, and X. Zhang. Estimating the confidence of conditional functional dependencies. In *SIGMOD*, 2009.

[6] W. Fan, F. Geerts, J. Li, and M. Xiong. Discovering conditional functional dependencies. *TKDE*, 23(5):683–698, 2011.

[7] J. A. Goguen and J. Meseguer. Unwinding and inference control. In *IEEE Symposium on Security and Privacy*, pages 75–87, 1984.

[8] H. Hacigümüs, S. Mehrotra, and B. R. Iyer. Providing database as a service. In *ICDE*, 2002.

[9] Y. Li, W. Dai, Z. Ming, and M. Qiu. Privacy protection for preventing data over-collection in smart city. *IEEE Transactions on Computers*, 2015.

[10] R. J. Miller, Y. E. Ioannidis, and R. Ramakrishnan. The use of information capacity in schema integration and translation. In *VLDB*, volume 93, 1993.

[11] M. Morgenstern. Controlling logical inference in multilevel database systems. In *IEEE Symposium on Security and Privacy*, 1988.

[12] M. Qiu, M. Zhong, J. Li, K. Gai, and Z. Zong. Phase-change memory optimization for green cloud with genetic algorithm. *IEEE Transactions on Computers*, 2015.

[13] N. P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography (PKC)*, pages 420–443. 2010.

[14] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 44–55, 2000.

[15] P. D. Stachour and B. M. Thuraisingham. Design of ldv: A multilevel secure relational database management system. pages 190–209, 1990.

[16] T.-A. Su and G. Ozsoyoglu. Controlling fd and mvd inferences in multilevel relational database systems. *TKDE*, 3(4), 1991.

[17] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):571–588, 2002.

[18] H. Wang and R. Liu. Privacy-preserving publishing microdata with full functional dependencies. *Data & Knowledge Engineering*, 70(3):249–268, 2011.