DMSD-FPE: Data Masking System for Database Based on Format-Preserving Encryption

Mingming Zhang[1], Guiyang Xie[1], Shimeng Wei[1], Pu Song[1], Zhonghao Guo[1], Zheli Liu[1], and Zijing Cheng[2(✉)]

[1] College of Computer and Control Engineering, Nankai University, Tianjin, People's Republic of China
zerinyl23@hotmail.com, xiao.sha.gu.a@l63.com, tunrekg@l63.com, wishercat@l26.com, l4l0587@mail.nankai.edu.cn, tunrekg@l63.com, liuzheli@nankai.edu.cn

[2] State Key Laboratory of Space-Ground Integrated Information Technology, Beijing Institute of Satellite Information Engineering, Beijing, People's Republic of China
linuxdemo@l26.com

Abstract. The traditional data masking systems cannot provide reversible operations for database, and they will destroy the referential integrity of database. To solve the problems above, we provide a new data masking system based on format-preserving encryption (DMSD-FPE). This paper presents the model of it and highlights the appropriate masking algorithms for different databases. DMSD-FPE could guarantee that the format of cipher text is the same as plain text, and provides reversible operations for databases. Besides, the referential integrity is also kept. Furthermore, the experiments demonstrates that the system is efficient enough to adapt to practical uses.

Keywords: Data masking · Data privacy · Format-preserving encryption

1 Introduction

Nowadays, we all expect the precise mining result from the massive data accumulated in the database. But how to protect the privacy of data becomes a new challenge. As we know, data masking is one of the solutions to this problem, which is a method to protect the privacy of desensitized data for the producing and testing environment. Currently, the public pay more attention on data masking and the application area becomes more than ever before, including the data mining based on privacy- protection in different test environments. For example, the hospital wants to dig out the dependencies between medicines and symptoms from the patient information database for a better treatment. In case of data loss or data leaks when the data is sent to a mining institution, the effective data mining method is needed.

With the increasing demand, requirements for data masking are becoming stricter and stricter. Databases contain much more sensitive information, such as ID number

J. Wan et al. (Eds.): CloudComp 2016, SPNCE 2016, LNICST 197, pp. 216–226, 2018.
https://doi.org/10.1007/978-3-319-69605-8_20

and address. Thus if the data was not masked in time, serious consequences like data leaks would be caused. Therefore, we must not only protect the sensitive information, ensuring the security of private data, but also meet the test environment and data availability requirements (data mining, statistical analysis, etc.). Meanwhile, for the particularity of data from database, the masking process should also guarantee the referential integrity.

As we know, the method of data masking is similar to that of data publishing, but there are some defects for traditional methods [9, 10]. One method requires to remove the sensitive attributes beforehand, which is unable to fully hold the original information. Another method needs to add redundant data to make it chaos, but the information could never be used as before. These methods are mostly one-way and irreversible masking strategy, which will break the referential integrity of database and can't meet the need for data privacy protection.

In addition to the foregoing methods, encryption is the most effective strategy for the protection of privacy. But encryption usually extends the data, such as AES and 3DES, which will output the ciphertext with the length of specified block size and make the ciphertext not to be stored in the original database anymore. Besides, the readability and usability of data become worse, which means, the data mining algorithms based on these data couldn't be used. Meanwhile, encryption will bring challenges to the database operation for ciphertext. It will destroy some of the operating characteristics of the plaintext, so that common queries and gathering operation are not allowed.

However, the format-preserving encryption (FPE [1–4, 11]) brings a new life to data masking for databases. It is necessary to encrypt the sensitive information of them, without breaking the referential integrity. The ideal way is to ensure that the ciphertext and plaintext have the same format (in the same domain), which is the method of FPE. Our data masking system based on FPE could easily protect data in case of data leaks, and it retains the available characteristic of data. Besides, the masking progress of the system is reversible.

1.1 Contribution

In this paper, we present a data masking system named "DMSD-FPE", which is based on format-preserving encryption (FPE). It adopts some data masking algorithms according to the needs of users. Compared with the traditional data masking methods, our system could keep the formation for different types of data and the referential integrity among data tables in the given database.

2 Our Proposed System

In this section, we will introduce where our system is applied to and how it works. Besides, in the second part, we will detail the system modules and explain the work principle of each module.

2.1 Application Scenarios

In practice, the data masking system is mainly applied in two areas, database backups and data mining.

Database backups: With the development of informatization in productive companies, the formal management of database is needed, such as database backups. If the backup database was stored in the form of plaintext, it would cause the information disclosure, so that data needs to be pre-masked before backing up. However, the data of the Shadow Database (the masked database) need to be decrypted sometimes. Then it requires the data masking process be reversible. Our DMSD-FPE system will be able to meet these needs.

Data mining: When the database holders want to send it to the public for data mining analysis, the important private information may be gotten by the third-party. In case of it, the holders could use DMSD-FPE system to mask the key attributes in advance.

2.2 System Model

The model of DMSD-FPE is shown in Fig. 1. It mainly works between the original database and the shadow one. This system could mask all information in the data tables of original database according to the users' decision, and insert the masked information to both new databases (Shadow Database) and new text files (Shadow Text). The information of shadow database is also available for analyzing and adopting in some types of data mining algorithms. We can get effective mining results both from the original database to the shadow one.

Fig. 1. System model

As shown in Fig. 2, DMSD-FPE includes three modules. Because there are three phases in the data masking system. We describe them with three modules in details as follows:

Setup Module: This module includes Source Database, Target Database and Information of Connection. We can choose the type of database, create target database, select the storage path of target database, and so on.

Rules Module: This module is the core operation part of the system, which sets all the rules for the data masking process. We could set key and algorithm for each data type in it, according to the table name and the attribute name. It mainly includes two components as follows:

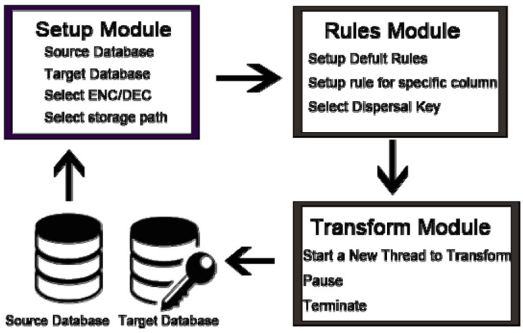

Fig. 2. Module structure of DMSD-FPE.

(1) *Key dispersion*: The DBMS-FPE only assigns one master key, and the encryption key for each column is dispersed via the key dispersal algorithm. This algorithm takes the table name and column name as input, with the dispersal one as output. In order to keep the referential integrity, we need to ensure that the key used for the primary key column and the foreign one is the same. So we must judge the relationship of all the tables first and generate the same key for them if necessary.
(2) *Masking algorithms*: DBMS-FPE will provide the corresponding masking algorithm according to the default data type or the user-chosen one. We will introduce each kind of algorithm in detail in Sect. 3.

Transform Module: In this module, we can start up the data masking process. And the system works in another thread with the masking algorithms chosen by us. After all work is done, the derived data will be imported into the target database via this module.

2.3 Work Flow

Overview. Data masking can be described as applying various basic methods and mixed ones to generate the similar data in the premise of satisfying the data constraints, through analyzing the raw data. Our system implements this process. To further describe this system, we introduce its work flows here:

Database Connecting: According to the type of databases provided by the provider, DMSD-FPE could connect to several common databases, such as SQL Server, Oracle, etc.

Data Analysis and Pre-processing: In order to obtain the structural information and constraints of databases, we should analyze the databases and find out the irrespective disturb. Then we can remove it via data pre-processing, as is called “Denoising”. And during the pre-processing, the system could generate the dictionary for the special data masking method (mentioned in Sect. 3.2).

Rules Specifying: This module is to set default masking rules for each column, according to original data types in the database. And also, the users could select the

rules (ID_ENC, FFX_INT_ENC, DATE_ENC, etc.) by their own. Finally, DMSD-FPE determines the corresponding data masking algorithms by the decided rules. It requires that the input data be read one tuple to another, so as the output.

Keys Selecting: During the key dispersing process, we'll get various keys for different columns. Therefore, the users may select different keys to mask the columns in one table. Each column to be masked could obtain one unique key. After that, the users need to save those keys in a file, so we could use the same keys to decrypt the data of the masking database directly.

Data Generating: The sub-procedure is to start masking the designated sensitive information tuple by tuple and generate the ciphertext. Then DMSD-FPE will import each ciphertext to a newly-build database (Shadow Database) and a text file (Shadow Text). It's much more convenient to publish or minethe Shadow Database and Shadow Text for effective results.

3 Data Masking Algorithms

In a DMSD-FPE, the core algorithm type is FPE. It mainly includes six kinds of algorithm. In this section, we will focus on the existing data masking algorithms, and introduce some new algorithms for special requirements.

3.1 Data Types and Corresponding Algorithms

In Table 1, there are some common data types and the masking algorithms adopted. We will introduce them in detail as follows.

Table 1. FPE for data types in database.

Data type	FPE schema
Integer	FFX_INT_ENC
Character	FFX_CHAR_ENC
ID number	ID_ENC
E-mail	EMAIL_ENC
Date	DATE_ENC
Items	Mixed schema

Traditional Data Type: For example, the integer [5] adopts FFX_INT_ENC to guarantee that the ciphertext value is within the specified range. The character string [3] stored in the database adopts FFX_CHAR_ENC, and the format of it is the length and storage size.

Expansion Data Type: Some data types, such as ID number, E-mail, Datetime [6] and so on, need to preserve the segment characteristic. So the masking process should base on corresponding algorithm for each of the segments. For instance, an e-mail number

consists of a customized string, a symbol "@", a domain name and the suffix. In order to preserve the formation of e-mail number, we could only operate the customized string. Based on the above, we adopt the ID_ENC, EMAIL_ENC and DATE_ENC severally for them.

As for the other special data types of the masking dataset, we call them items, which adopts ITEM_ENC algorithm to do the masking job. This algorithm takes the mixed masking mode, including FFX_INT_ENC and other method for the traditional data type. Let's take the medical data as an example, the ITEM_ENC for it is just an inner substitution method. To preserve the item's format, we should only ensure that the ciphertext belongs to the same dataset as cleartext.

3.2 Implementation Details

To preserve the association rules of the original database and to meet the need of masking sensitive properties, DMSD-FPE mainly adopts six masking methods introduced in 3.1. Here we will introduce the pseudo-code of the main masking algorithms:

(1) Algorithm for Integer

The pseudocode of this algorithm is shown in Table 2. Before masking the integer data, we need to preprocess it. Firstly, we've to check the ASCII table to find out the value of each integer character, and transform the input string into a decimal integer. Secondly, the integer should be divided into two. The right part(R) is transferred to the left. Then we should deal with R through AES algorithm and XOR the result with the left one (L) as the new R'.

Table 2. Pseudocode of FFX_INT_ENC

```
Input:OriginalData, Maxvalue
1:  For  i=1 to 6 do
2:OriginalData→left || right
3:right→Dest_left
4: AES_KEY (right)⊕left→Dest_right
5: Dest_left   || Dest_right→OriginalData
6:  End for
7: OriginalData→EncryptData
8:  If  EncryptData<Maxvalue
9:    FFX_INT_ENC(EncryptData , Maxvalue)
10: End If
Output:  EncryptData
```

The process above is based on cycle-walking [8], which will be executed for six times all in all. And if the result larger than the maximum value, the transformation will be repeated until it is less than the maximum value.

(2) Algorithm for Character String

The pseudocode of this algorithm is shown in Table 3. In the process of masking the character data, we need to set up a character type dictionary (covering all possible characters). When inputting a string, the program would check up the dictionary to find out the index of each letter and transform the input string into a decimal integer. The rest of the process is similar to FFX_INT_ENC, but the difference is that there is no need to do the cycle-walking.

Table 3. Pseudocode of FFX_CHAR_ENC

```
Input: OriginalData
1: CharToNumber(OriginalData)→original
2:  For i=1 to 6 do
3:    original→left || right
4:    right→Dest_left
5:    AES_KEY (right)⊕left→Dest_right
6:   Dest_left || Dest_right→original
7:  End For
8:  NumberToChar(original)→EncryptData
Output: EncryptData
```

(3) Special methods

The pseudocode of this algorithm is shown in Table 4. Usually, some special data set has a fixed property, and the masking process can't change the characteristic to the original dataset. It is only the substitution of the existing dataset elements. For example, one product's name should be masked to another product's name.

Flow: In the process of implementing this algorithm, we need to get all the elements of an attribute A and generate a dictionary (Dictionary) of attributes to be queried. Check up the index of element E in the dictionary, and then mask the index to get an index' in a specified range via FFX_INT_ENC. The element in Dictionary whose subscript is index' is the ciphertext A'.

(4) Summary

All of the data masking algorithms mentioned above can preserve the formation of data, and won't influence the availability of data. When we implement different algorithms, we can get different keys through dispersing MK, so that the results are decentralized and the difficulty of cracking is also increased. When returning the mining results, we can also decrypt the data via MK or the key dispersion algorithm.

Table 4. Pseudocode of ITEM_ENC

```
Input: Atrribute A, Originallist
1: Foreach (Table in DataBase)
2:   GetDictionary(Table, A)→Dictionary
3: End foreach
4: For i=1 to Originallist.count do
5:   If Originallist[i]==Dictionary[j] then
6:                          k=FFX_INT_ENC(j,length,
encryptone,maxIndex);
7:     EncryptList.Add(Dictionary[k]);
8:   Else
9:     EncryptList.Add(Dictionary[j]);
10:  End if
11:End for
Output: EncryptList
```

4 Experiment

In this section, we will focus on some details of implementing our data masking system. And also, we'll consider the data analysis, such that evaluate the efficiency of the data masking algorithms.

4.1 Implementation Details

We implement our DMSD-FPE system by Windows C#, an object-oriented programming language. And we implement all of the algorithms through an open kernel API of C++ DLL, called "FPEDLL". As a shared DLL file, it can be invoked by the system program to mask the data. Developers could build other data masking systems based on this DLL with the public interface.

In our Windows Forms application (FPE), we encapsulate five classes totally, including DBFactory, DBOperation, EncryptSelect, DecSelect, ThreadMethod and so on. The functions are as follows:

- DBFactory class: it is used to select the type of database, such as SQL Server, MySql, etc.
- DBOperation class: it contains the operation code of various types of database, including Create, Read, Update and Delete (CRUD).
- EncryptSelect class: it is used to select the encryption method, which calls the encryption algorithms in FPEDLL.
- DecSelectSelect class: it is used to select the decrypting method, which calls the encryption algorithms in FPEDLL.
- ThreadMethod class: it is a class of thread, used to display the progress of encryption and decryption. It contains some data masking process and database operation.

4.1.1 User Interface

The following pictures are the main interfaces of our data masking system, DMSD-FPE, which are shown in Fig. 3.

Fig. 3. Interface of DMSD-FPE

The function of the modules in the interface were introduced in detail in Sect. 2.2, and at the bottom of each interface there are some notes to explain the usage of them.

When the users click the Next-Step Button, they should choose whether or not to save the alteration of the current page, which ensures that each action is always right.

What's more, when clicking on the Start Button, the progress bar will show the current processes.

4.2 Efficiency

We performed all of the experiments on Computer of Inter(R) Core(TM) i7-4510U CPU @ 2.00 GHz with Windows10 OS. The efficiency of general data masking algorithms mentioned in Sect. 3 is shown as the follow one (Table 5).

Table 5. Execute time of algorithms

Records number	10	100	500	1000	5000	10000
FFX_INT_ENC	0.0426 ms	0.39420 ms	1.6679 ms	3.3731 ms	17.2152 ms	35.2001 ms
FFX_CHAR_ENC	0.1580 ms	1.4971 ms	6.3412 ms	14.1846 ms	74.9092 ms	137.5970 ms
ITEM_ENC	0.0031 ms	0.1445 ms	3.1749 ms	9.4949 ms	11.7808 ms	13.0087 ms

As shown in Table 5, for the first two encryption algorithm, the execute time has linear relationship with the number of records, which increases exponentially with the number.

However, there is not distinct regularity of execute time for ITEM_ENC algorithm. Because in the process of masking with this algorithm, we need to traverse the items dictionary to find out the index of each item, and the time for each one is actually different according to different location.

As shown in Table 6, the time of data masking is almost not affected by the number of the tables. It only depends on the number of records in the source database, which shows a linear relationship. In addition, the time to create different databases is basically equal, and only becomes more when the number of tables increases.

Table 6. Time of exporting metadata for database tables

Source database		Time-consuming	
Table number	Record number	Data masking	Database creating
1	25	0.4971 s	0.7182596 s
2	25	0.5047 s	0.8073632 s
3	25	0.5091 s	0.8196084 s
1	50	0.9048 s	0.6484303 s
2	50	0.7512 s	0.7388628 s
3	50	0.9063 s	0.7462538 s
1	75	1.3392 s	0.8196084 s
2	75	1.4026 s	0.8816251 s
3	75	1.5477 s	0.7208608 s

5 Conclusion

In this article, we put forward data masking system for database based on FPE. For different types of database we provide suitable masking algorithm according to their shelter needs. We can keep data formats and referential integrity of the database without removing sensitive information at the same time. We implemented this system through experiments. Experiments show that efficiency can meet the needs of practical applications and encryption, moreover, decryption process is reversible.

In order to verify the correctness of this masking system, we still need to use the existing association rule mining and other data mining algorithms to do mining analysis of information for masking database. We look forward to coming to a conclusion, which is the same as the mining result from the original database, to guarantee the correctness of this system.

References

1. Black, J., Rogaway, P.: Ciphers with arbitrary finite domains. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 114–130. Springer, Heidelberg (2002). doi:10.1007/3-540-45760-7_9
2. Morris, B., Rogaway, P., Stegers, T.: How to encipher messages on a small domain. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 286–302. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03356-8_17
3. Bellare, M., Rogaway, P., Spies, T.: The FFX mode of operation for format-preserving encryption. NIST Submission

4. Hoang, V.T., Morris, B., Rogaway, P.: An enciphering scheme based on a card shuffle. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 1–13. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32009-5_1
5. Terence, S.: Feistel finite set encryption mode. NIST Proposed Encryption Mode
6. Liu, Z., Jia, C., Li, J.: Format-preserving encryption for datetime. In: Intelligent Computing and Intelligent Systems, vol. 2, Springer (2010)
7. Li, M., Liu, Z., Li, J., Jia, C.: Format-preserving encryption for character data. J. Netw. **7**(8), 1239–1244 (2012)
8. Li, J., Jia, C., Liu, Z.: Cycle-walking revisited: consistency, security, and efficiency. Secur. Commun. Netw. (2012)
9. Espositoa, C., Ficcob, M., Palmierib, F., Castiglionec, A.: A knowledge-based platform for big data analytics based on publish/subscribe services and stream processing. Knowl.-Based Syst. **79**, 3–17 (2015)
10. Yang, J.-J., Li, J.-Q., Niu, Y.: A hybrid solution for privacy preserving medical data sharing in the cloud environment. Future Gener. Comput. Syst. 43–44 (2015)
11. Spies, T.: Format Preserving Encryption. Voltage White Paper