2018 10th International Conference on Knowledge and Systems Engineering (KSE 2018)

Ho Chi Minh City, Vietnam
1-3 November 2018

IEEE Catalog Number: CFP1803I-POD
ISBN: 978-1-5386-6114-7

Computer Vision and Pattern Recognition

Natural Language Processing and Text Mining

Key-value based data hiding method for NoSQL database

Ta Minh Thanh*, Nguyen Huu Thuy*, Ngoc-Tu Huynh†
*Le Quy Don Technical University, 236 Hoang Quoc Viet, Ha Noi, Vietnam.
Email: thanhtm@mta.edu.vn, nguyenhuuthuy91@gmail.com
†Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh city, Vietnam.
Corresponding author, Email: huynhngoctu@tdtu.edu.vn

***Abstract*—This paper proposes a key-value based data hiding method for NoSQL database. Our proposed method is applied on the database generated by MongoDB (Document-Oriented Database). It only modifies the structure of the key-value attributes for hiding information without affecting the contents stored in the database. On technical aspects, our method allows flexibly to change the sequence of the key-value attributes in order to embed the bit "0" or bit "1" into the rows of tables in the database. After hiding the information into the database of MongoDB, the operations such as $select$, $insert$, $export$, $import$ and so on.**

***Keywords*—Key-value Database, NoSQL, Document-Oriented Database, Data hiding.**

I. Introduction

A. Background

Digital watermarking is the promising technique for protecting the copyright of digital contents. In this solution, the watermark information $e.g$ logomark, random number sequences, is embedded into the digital contents in order to prove the ownership in case of any dispute. The watermark information can be extracted from the watermarked contents and compare with the original watermark registered by copyright authority. Note that, the quality of watermarked content should be imperceptible to maintain image or video quality. Almost of the watermarking methods are proposed to apply on the digital multimedia.

Today, every services are developed by web-based applications and deployed on web servers. The valuable data of those services are stored in the database servers. Therefore, we consider to protect the copyright of the web data resides on the database servers. According to our survey, most of proposals for database watermarking is mainly to focus on the copyright protection problem of relational databases. Although the relational database is widely used in the web applications, it requires developers and applications to strictly structure the data used in their applications. In the structure of relational database, the data is stored in highly structured tables with predetermined columns of certain types and many rows of the same type of information. Therefore, if the developers use many-to-many relationships in the database, it may cause the low performance for web applications with the bigger and bigger data stored in database nowadays.

In order to solve the highly restricted conditional of relational database, NoSQL database is proposed for improving the performance and making more flexible. Therefore, NoSQL guarantees to be used in big data and real-time web applications [1]. Developers employ NoSQL database for their web applications because of its simplicity of design, simpler "horizontal" scaling to clusters of servers.

Databases are a part of applications such as websites, programs, and so on. Users go through websites or programs to access the database and get the information they need. In fact, NoSQL is usually stored as key-value pairs, or key-value databases, implements a simple data model that pairs a unique key with an associated value (stored in a format similar to JSON). Since the model of NoSQL is simple, it can lead to the development of key-value databases, which are extremely performant and highly scalable for session management and caching in web applications. Such kind of NoSQL databases make the amount of stored data more and more and valuable for real applications.

Since the big data can be stored into the NoSQL by using the structure of key-value, we also can control the stored structure of NoSQL database to hide the secret information. The embedded information can be used for authentication, secret message transmission, copyright protection, and so on. To the best of our knowledge, the researches of data hiding into NoSQL databases, for instance $i.e$ MongoDB, are still limited.

B. Our contributions

In this paper, we realize that the need for data hiding of NoSQL to deter data piracy and to authenticate have been required. In this paper, we propose a data hiding method for NoSQL database server.

Our paper focuses on the following characteristics for controlling the sequence of the attributes of MongoDB that does not effect the meaning of information stored in the database. Our contributions are listed as follows:

(1) We propose a new data hiding method by using the characteristics of MongoDB that even changes the order of the attributes in the structure, the value of the stored data information is remained.

(2) After changing the order of the attributes of the database for information embedding, the results of the query information via web applications are not changed. Based on these characteristics, we also propose a method using the combination of the attributes of the database to make embedding pattern for hiding the confidential/secret information into the data structure of

978-1-5386-6113-0/18/$31.00 ©2018 IEEE

MongoDB without being detected due to structural changes made by the software.

(3) Using our proposed method, the MongoDB database can be normally exported/imported by the software. When the web applications query to the database, only the results of the query information are shown without displaying the order of attributes. Therefore, our method is effectively applied on NoSQL databases.

C. Roadmap

The remainder of this paper is organized as follows. Section 2 introduces the preliminary background and related works. Next, the method of hiding information in a NoSQL data file by combining attributes of a database is proposed in Section 3. Section 4 shows evaluation of the confidentiality and security of the proposed method. Finally, Section 5 summarizes the results and gives the next research direction.

II. Preliminary background and Related works

A. Preliminary background

Although NoSQL (Not Only SQL) databases are used widely in web applications, the solutions for protecting the database itself are still limited. Therefore, this paper proposes the key-value based data hiding method employing the combination of attributes from the table of databases.

B. Related works

Most of watermarking methods proposed for database focused on the relational databases. It is difficult to apply on the non-relational database such as MongoDB.

The early proposal method [2] had shown that the copyright information can be embedded into the relational databases by using some selected attributes out of several candidates attributes in a tuple. The watermark information is embedded into the LSB (Least Significant Bit) of a selected attribute. The LSB of specific bit locations of attributes is defined by using the control of a secret key generated for only owner of the database. However, the embedding method of [2] is simple and it is also not resilient under database attacks such as randomizing of the LSBs of random attributes. The paper of Khanduja $et.\ al.$ [3] had proposed a watermarking method for relational database using bacterial foraging algorithm (BFA). The order of tuples in database is modified for embedding the watermark bits. Their paper uses BFA for turning to reduce the execution time. Their method also must remain the relationship condition of database during the embedding and extraction process. By another idea, Khanduja and Verma [4] had proposed the embedding for relational databases based on randomly selected multiple attributes in order to insert the watermark bits into the LSBs of attributes. Another papers [5], [6], [7] have contributed to embed the watermark into the relational database. Those proposal methods always require to adjust the LSBs or re-oder the tuples of the selected attributes for embedding the watermark bits. Especially, in the embedding methods of relational databases, the researchers must pay their attention to remain the relationship between the tables, attributes, primary keys and so on of the database. If the relationship of the relational database is destroyed during the embedding process, the database cannot be used in the web applications.

The proposal watermarking method for NoSQL database [8] is also presented recently. This embedding method employs the feature of NoSQL which is flexibly increasing and often have irregularities in schema, for inserting the watermark information. In order to embed the watermark information, a redundant attribute is flexibly injected into the tuple and the watermark information is stored into the redundant attributes. Although their method can resist the slight modification attacks, the redundant attributes increase into the database. It may effect to the performance of web applications when the data is growing up more and more.

A fragile zero-watermarking [9] is proposed for detecting and characterizing malicious modifications in database relations. In this method, the watermark information is not embedded into the relational database itself. They had proposed an idea that generates the watermark information by using the local characteristics of database relation such as frequency distribution of digit, length and range, an so on. However, this method is needed to register with trusted third party. When the dispute happens, the reference from trusted third party is required for judgement.

According to our survey of prior works, we realize that most of prior works tried to propose the watermarking method for relational database. Although the need of NoSQL database increased, the focusing on the copyright protection of NoSQL database is still limited. Therefore, our paper presents a new method for embedding the watermark information into the NoSQL database. Specifically, we focus on the key-value based NoSQL database such as MongoDB.

III. Key-value based watermarking method

We employ the features of key-value NoSQL for evaluate our proposed method. Our method does not change the LSBs of selected attributes. Our method also does not insert the redundant attributes to embed the watermark information. We change the order of selected attributes to present the watermark patterns.

A. Characteristics of MongoDB database

MongoDB is an open-source database, which have high performance, high availability, and can automatic scaling [10].

A record in MongoDB is a document, which is a set of key-value pairs (shown in Figure 1). It have dynamic schema, documents in the same collection do not need to have the same structure. The values of fields may include other documents, arrays, and arrays of documents. It can hold many types of data and similar to JSON objects [10].

MongoDB documents are store in collections as shown in Figure 2. Collections are the same as tables in relational databases [10].

The stored documents in MongoDB are a set of key-value pairs. Therefore, it is similar to JSON objects, so it has a great dynamics schema. Based on the characteristic of the stored documents, we can apply the hiding method by

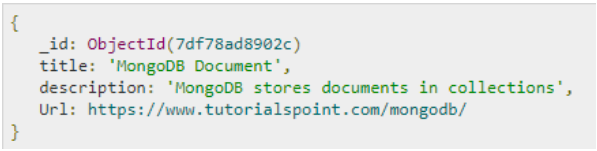

Fig. 1: Key-value based storage in MongoDB.

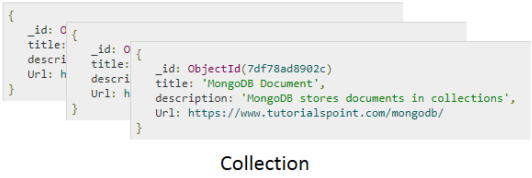

Fig. 2: MongoDB stores documents in collections[10].

using the attributes of MongoDB database efficiently. Those characteristics are explained as following.

(1) After changing the order of the attributes (keys) of MongoDB database, the results of database queries from the web-based application such as $select$, $insert$, $update$, $create$, are not affected. Users of web-based applications cannot detect the changing of key-value structure of MongoDB database. It is transparent for front-end users.

(2) After changing the order of the attributes (keys) of MongoDB database, the administrator can export the MongoDB database from the web server and import it into another web server without the affection of stored data. That means we can change flexibly the key-value structure of database and export/import it to distribute it via Internet.

In this paper, we employ the characteristics of key-value database for embedding the secret information into the structure (the order of attributes) of database. We exploit (1) to generate the order patterns of key-value according to the watermark patterns. The order patterns of key-value also are used as the secret key for extraction the secret information. After embedding the secret information, we can employ (2) for distributing the embedded database into another database servers. The secret information can be extracted for the purpose of copyright protection, ownership proof, an so on.

In order to show the characteristic (1), we generate a sample database of MongoDB, then change the order of attributes of data structure in database. As shown in Figure 3, the order of attributes in the top record and that of the bottom record are different. In the key-value based database, the different order of attributes in records are allowed. In this example, the attributes of the database such as Age, Phone, National, LastName, FirstName, Mobile, Sex are used in a different order. However, when the query for collecting the data information from the database server, the results are not altered when comparing with the original data. That means even change the order of attributes of the record in MongoDB database, it does not affect the performance of web-based applications.

According to those characteristics, by changing the order of attributes of MongoDB database, we can hide more bits of secret information into the key-value database. The order of the attributes of the database that will not affect the results

Fig. 3: Example of sequential attribute of Document.

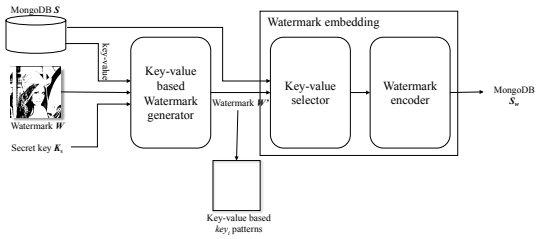

Fig. 4: Our proposed embedding method.

when we make queries to the database. Therefore, the proposed method can be used to hide the secret information in the NoSQL database, especially for MongoDB database.

B. *Proposed key-value based data hiding method*

Our proposed method consists of two processes: the embedding method and the extraction method. In the extraction method, the original MongoDB is not required in the extracting process.

1) Embedding method: In the embedding method, the original MongoDB S, the secret K_s, and the original watermark W are required. The watermark information W is necessary to be scrambled by the key-value (attributes) of S in term of increasing the security. That makes the watermarks patterns W' for embedding process. Afterwards, W' is embedded into the S. By doing so, the embedded MongoDB S_w is generated.

The embedding method can be shown in Figure 4. We summarize the main points of our proposed embedding method as follows:

(1) All key-value $\{(key_i, value_i),\ i = 1, \cdots, N\}$ are extracted from S. N is the number of key-value created in S. The secret key K_s defines the length of watermark pattern W' generated from the watermark information W. $K_s \leq N$ is also the security parameter in order to enhance the robustness of our proposed embedding method.

(2) In the key-value based watermark generator process, the number of selected keys $M = K_s$ of key_i are re-ordered based on the watermark patterns of W. That process makes the W' for embedding.

(3) In the process of embedding, the key-value selector chooses the appropriate order of key_i (attributes) for embedding. According to this process, the key-value based key_i patterns tables is saved for extraction process.

(4) After choosing the order of key_i, the watermark patterns W' are encoded into the key-value of S, then S_w is created.

Table I shows a sample of watermark generator and key-value selector used in our proposed method. Watermark patterns (M bits) construct all sequences of key_i (attributes);

TABLE I: The sample of watermark patterns used in our embedding method

Embedded M bits patterns	**Selected key_i (Attributes)**			
0000	Age	National	Sex	FirstName
0001	Age	National	FirstName	Sex
0010	Age	Sex	National	FirstName
0011	Age	Sex	FirstName	National
0100	Age	FirstName	Sex	National
0101	Age	FirstName	National	Sex
0110	National	Age	FirstName	Sex
0111	National	Age	Sex	FirstName
1000	National	Sex	Age	FirstName
1001	National	Sex	FirstName	Age
1010	National	FirstName	Age	Sex
1011	National	FirstName	Sex	Age
1100	Sex	Age	National	FirstName
1101	Sex	Age	FirstName	National
1110	Sex	National	Age	FirstName
1111	Sex	National	FirstName	Age
unused	Sex	FirstName	Age	National
...	...	...	...	
unused	FirstName	Age	National	Sex

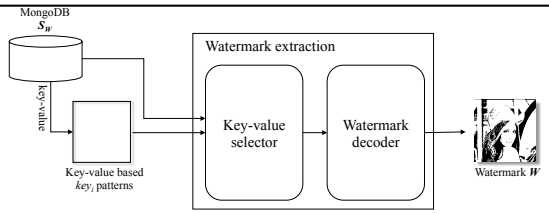

Fig. 5: Our proposed extraction method.

simultaneously, indicate the sequence of bits that are hidden in each combination. The initial state of the key_i combination will be used to hide the bit string information “0000". The “unused" combinations will not be used for embedding information in the structure of S.

2) Extraction method: In the extraction process, only the watermarked MongoDB database S_w and key_i patterns tables are required. The steps of extraction method can be shown in Figure 5. Those steps are explained as follows:

(1) All key-value $\{(key_i, value_i),\ i = 1, \cdots, N\}$ are extracted from S_w.

(2) Based on the key_i patterns tables, the selected order of key_is are specified.

(3) In the process of extraction, according to the key-value selector, the watermark patterns are retrieved from many records of MongoDB database.

(4) Collect all watermark patterns extracted from the database, the original watermark can be reconstructed.

After reconstructed the watermark, it is used for copyright proof, adjustment detection, and so on.

An example of extraction watermark pattern from a record of MongoDB when used the Table I is shown in Figure 6. According to the Table I, the watermark pattern is extracted from the embedded record is “0001".

C. Capacity


```
{"_id":{"$oid":"58a8622aec296570e9cca32a"}, "Age":
53.0, "National":"United Kingdom", "FirstName":"Phan"
, "Sex":"2"}
```


Fig. 6: Example of extraction watermark pattern from a record.

Assume that we select M key-value (attributes) from N key-values of S for hiding information to create the key_i patterns tables. In the $M!$ combination of attributes, $2^{log_2(M!)}$ orders can be used to hide the message. Therefor, each order of key-value (attributes) can hide the number of watermark bits as follows:

$$log_2(M!)[\text{bits}]$$

If in the original MongoDB database S, the number of occurrences of the document has an attribute order was selected as the key_i patterns tables was N times, the total amount of confidential information hidden in embedded database S_w is:

$$N * log_2(M!)[\text{bits}]$$

D. Security

The decisive point to create a watermark key_i patterns table is the sequential appearance of the key-value attribute in the original database. This sequence determines the first bit pattern hidden as "000 ... 0". This characteristic also determines the robustness of the tidbits by the probability of the association for the attribute in the original database is huge. If the key-values have N attributes, then the sequential properties of the key-value in the a record will be $N!$ arrangement. Only when the sequences of attributes in the original database are determined, it is possible to extract confidential information from the embedded database.

On the other hand, in the key_i patterns table, all combinations of attributes are not used to hide the message. $N! - 2^{log_2(M!)}$ remaining combination (marked "unused") will be the probabilities that make it difficult to attack confidential data analysis in embedded database.

IV. Experimental Results

In this section, we explain the results of the experimental study that analyzes the resilience of the our watermarking scheme to some attacks. All the experiments were performed on 2.1 GHz Intel Core i5 CPU with 12GB of RAM. Our experimental results can be applied on the database of the US [11].

A. Performance analysis

In order to better understand, we used the secret key $K_s = M = 4$ for generating the embedded watermark patterns. The watermark image is Girl image that is converted to binary image as shown in Figure 7 with size of $16 \times 16, 32 \times 32, 64 \times 64, 128 \times 128, 256 \times 256$, respectively. We surveyed the necessary number of records for embedding all watermark image. The necessary number of records used to embed all watermark patterns of Girl image with size of $16 \times 16, 32 \times 32, 64 \times 64, 128 \times 128, 256 \times 256$ are 126, 190, 574, 2110, 8254 records, respectively.

According to the necessary number of records for hiding all watermark image, we also surveyed the hidden time and

Fig. 7: Watermark image.

TABLE II: Experimental results table

Database size (Megabytes)	Total Capacity (bits)	Hidden Time (Seconds)	Extracted Time (Seconds)
10	26000	78.94	25.59
20	52000	88.78	50.74
50	130000	109.02	130.70
100	260000	127.87	271.45
150	390000	180.78	356.53
200	520000	222.11	459.89
250	650000	255.31	592.08
300	780000	289.13	623.61
350	910000	344.97	708.79
400	1040000	438.67	825.27
450	1170000	497.41	889.33
500	1300000	553.45	968.76

the extracted time based on the size of MongoDB database. We generated the MongoDB database by increasing the key-value of that and importing the dummy data into the database. The experimental results are shown in Table II. It is clear that the hidden time, the extracted time, and capacity are proportional to the size of MongoDB. It also showed that our proposed method obtained the efficient way to hide the secret information into the MongoDB databases.

B. Analysis of robustness

This paper embeds the watermark patterns into the order of key-value of MongoDB databases. Thus, altering the values of records or increasing more keys (attributes) into the database structure will have negligible effect on watermarking bits patterns. Therefore, this makes our algorithm robust against various content-based attacks to the embedded MongoDB databases such as altering values of records, inserting more key-value, randomization, zero out and bit flipping attack for relational databases.

We were care about the records removing attack applied on the MongoDB. If the records in the databases are removed, the embedded watermark patterns may be removal. However, in this proposed method, we repeatedly embedded the watermark logo into the MongoDB databases, therefore, although attacker removes several records from the databases, it does not effect more on extracted watermark logo. In order to prove that assuagement, we attacked the watermarked database by removing random records into the database. The experimental results are shown in Figure 8. We randomly removed some records of databases. The number of removal records is 50, 75, and 100, respectively. According to the results of Figure 8, the extracted information is not affected so much. Since our proposed method repeatedly embeds the watermark into the MongoDB databases, thus, it makes robust against removal attack.

V. Conclusion

This paper has presented an algorithm that uses the key-value of the MongoDB databases to represent the hidden information using the watermark patterns. By changing the

Fig. 8: Example of records removing attack.

sequential nature of the key-value (attributes), the content of the hidden database is not affected and the structure of the database is not significantly disturbed. On the other hand, due to the large number of document attributes the number of combinations produced is relatively large, leading to the complexity of the cryptanalysis.

References

[1] NoSQL: https:en.wikipedia.orgwikiNoSQL

[2] Agrawal R, Haas PJ, Kiernan J, "Watermarking relational data: framework, algorithms and analysis," VLDB J 12(2), pp. 157–169, 2003.

[3] Khanduja V, Verma OP, Chakraverty S, "Watermarking Relational databases using Bacterial Foraging Algorithm," Multimed Tools & Appl.;74(3):pp. 813–839, 2013. DOI: 10.1007s11042-013-1700-9.

[4] Khanduja V, Verma OP, "Identification and proof of ownership by watermarking relational databases," Int J Inf Electron Eng 2(2):pp. 274–277, 2012.

[5] Farfoura ME, Horng SJ, Lai JL, Run RS, Chen RJ, Khan MK, "A blind reversible method for watermarking relational databases based on a time-stamping protocol," Expert Systems with Appl. 39(3):pp. 3185-3196, 2012.

[6] Ifthikar S., Kamran M. and Anwar Z. RRW-A robust and reversible watermarking technique for relational Data. IEEE Transactions on Knowledge and Data Engineering. 27(4):pp. 1131-1145, 2015. DOI: 10.1109/TKDE.2014.2349911.

[7] Chen X, Chen P, He Y, Li L, "A self-resilience digital image watermark based on relational database," Int Symp Knowl Acquis Model, pp. 698–702, 2008.

[8] Khanduja V, "Dynamic Watermark Injection in NoSQL Databases," J Comp Sci Appl Inform Technol. 2(1): 5, 2016. DOI: http://dx.doi.org/10.15226/2474-9257/2/1/00108.

[9] Khan A, Husain SA, "A fragile zero watermarking scheme to detect and characterize malicious modifications in database relations," The Scientific World Journal, pp. 1–16, 2013.

[10] "The MongoDB 3.4 Manual," https:docs.mongodb.commanual.

[11] National geochemical survey database of the US, http:tin.er.usgs.govgeochem