International Journal of Security and Its Applications
Vol.8, No.6 (2014), pp.25-40
http://dx.doi.org/10.14257/ijsia.2014.8.6.03

Block-Based Scheme for Database Integrity Verification

Lancine Camara[1,2], Junyi Li[1,2], Renfa Li[1,2], Faustin Kagorora[1,2] and Damien Hanyurwimfura[1,2]

[1]*College of Information Science and Engineering, Hunan University, Changsha 410082, Hunan, China.*
[2]*Key Laboratory for Embedded and Network Computing of Hunan Province, Changsha, Hunan 410082, China.*
[1,2]*lcamaralancine@yahoo.fr*, [1,2]*junyilee@hnu.edu.cn*, [1,2]*lirenfa@vip.sina.com*, [1,2]*kagororaf@yahoo.com*, [1,2]*hadamfr@yahoo.fr*

Abstract

Databases play an important role today in every modern organization, verifying their integrity is needed. Watermarking can be used to protect the integrity of database. In this paper, we present a secure fragile embedding watermark technique to verify the authenticity of an outsourced numeric relational database. Our technique treats the watermark embedding as an optimization problem by securely inserting a single watermark bit in individual database partition and the optimal threshold is computed for watermark detection. The approach partitions the database in different groups of square matrix and modifies the database while preserving the field values usability constraints. The database group determinant value is used to compute the position of field to be marked. Furthermore, we evaluated our scheme on a real case study and results show its effectiveness. The proposed scheme can detect and localize the malicious modifications made to the database. The proposed technique is highly resilient to common attacks and it overcomes some limitations of previous approaches on fragile watermarking.

Keywords: *Database Watermarking, Integrity Verification, Optimization problems, Determinant*

1. Introduction

Databases are ubiquitous, and it is significant to protect them against piracy, illegal claim of ownership, and the use of un-reliable database. To reach this goal, database watermarking techniques are commonly used. From the literature reviews, database watermarking can be classified as robust or fragile depending on the purpose. For database watermarking in general whether it's robust or fragile one requirement for the security purpose is the use of a secret key. The robustness of watermarking technique [1] is used in practice for ownership proof. The robust scheme reaches it aim if the watermark can survive after any malicious attacks made to the database with the aim to remove the watermark.

The watermark or simply the mark can be considered as a sequence of binary bits to be inserted in the database. Recently some robust watermarking techniques insert the mark in database by preserving the database information [2-4]. A multimedia fingerprinting technique [5, 6] is a robust watermarking technique used to identify individual multimedia distributed copy; it is robust against any attack attempt to remove the watermark. Whereas the fragile watermarking technique [7, 8] is generally used for digital data content integrity verification, the watermark must be destroyed after any malicious modification made to the database. To

ISSN: 1738-9976 IJSIA

allow some legitimate modification on the database while conserving the watermarking fragile, semi fragile watermarking [9] was proposed and became interest of peace of research. The watermarking techniques described above mainly focused on numerical database, some watermarking techniques [10-12] are used on non numerical database.

From the literature, many pieces research focus on robust watermarking technique than fragile watermarking technique, we don't know the motivation behind this interest but fragile watermarking is important. For example, consider tampering someone's data in any domain (business, education, *etc.*,), and some military or medical databases are very sensible to modification. The consequence of such act may not be acceptable, we believe that it is time to give more importance to the verification of databases authentication simply in fragile watermarking approach. . Most of the works on fragile database watermarking developed to assure the authentication of database presented one or both of the following limitations: (i) the field values are altered without considering the data usability constraints (ii) simple tuples or attributes sorting may be considered as malicious attacks. This paper focuses on the aspect of the authentication verification of numeric database with an extension to non numeric database.

The main contributions of this paper are:

We propose a novel partitioning approach that enables the use of square matrix determinant to verify the authentication of database.

The presented approach minimizes the amount of alteration in database by inserting a single bit in one field of each group by preserving the field data usability.

Finally, we evaluated the approach on a Forest Cover Type dataset [13], a real database from US Geological Survey and US Forest Service data to prove the resilience of our approach against various attacks.

Our approach mainly focuses on matrix determinant computation. An overview of square matrix and it determinant are showed below.

A matrix is a rectangular array of numbers. A matrix with *m* rows and *n* columns is called an *m* x *n* matrix [14]. The matrix may contain complex numbers and each number that composes the matrix is called an element of the matrix. Two matrices are equal if they have the same number of rows and the same number of columns and the corresponding entries in every position are equal. A matrix B is called square matrix if it has the same number of rows as columns and represented by:

$$B = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \quad (1)$$

The determinant of a square matrix "B" denoted by "det B" or simply |B| is the real number that can be computed from the square matrix B, its computation formula is given in (1). The minor of any element a_{ij} of a square matrix "A" denoted M_{ij} is the determinant obtained when the row and column of that element are deleted. Let "B" be a square matrix of order n with coefficients in R , the determinant of A is computed as follows:

$$|A| = \sum_{i=1}^{n} (-1)^{i+1} a_{i1} \det(A_i) \quad (2)$$

Where A_i is obtained by deleting the first column and i^{th} row.

This paper discusses issues that arise in the fragile database watermarking; it describes an approach to overcome the above limitations. This paper is organized as follows. The section 2 reviews the related works. In section 3, an overview of our technique is explained

and each step is described. Section 4 describes the details of our approach. Section 5 analyzes and presents the results of our experiments. The security and scalability of our approach are discussed in section 6. Section 7 concludes our paper with highlight to on-going work.

Table 1. Notation and Parameters

Symbol	Description
$\nu = \frac{\alpha}{\gamma}$	The number of partitions (or groups) with α the number of tuples and γ the number of attribute
W	The watermark binary string
l	The length of watermark
S_j	The j^{th} partition (or group)
$r_i.p_k$	i^{th} primary key tuple
K_S	The secret key
Δ_i	The data usability constraints
$\tau = \{\max_{S_j}, \min_{S_j}\}$	The group embedded maximization and minimization determinant values
τ^*	The optimal decoding threshold

2. Related Works

During the last decade, database watermarking attracted much attention. The pioneer work in database watermarking technique was proposed by Agrawal, *et al.*, [15], they identified the least significant bits (LSB) position of marked tuples and attributes to embed the watermark bits. The watermark is easily lost by shifting the LSB bit position. Therefore embedding the watermark in LSB position may not guaranty the data usability constraints. In [16], a robust group based watermarking technique resilient to common malicious attacks (insertion, deletion, and alteration) is proposed, the watermarking process is formulated as constrained optimization problem, the genetic algorithm (GAs) and pattern search are used to solve the problem. They used optimal threshold to detect the watermark. Kamran, *et al.*, [4] proposed a robust watermarking approach highly resilient to malicious attacks. Their embedding approach is subject to soft usability constraints while allowing tight usability constraints that minimize the distortion in the data and their watermark decoding technique is independent to usability constraints. Schachtner, *et al.*, [17] proposed a determinant criterion to constrain the solutions of non-negative matrix factorization problems to provide an exact solution. Shah, *et al.*, [11] proposed a distortion free and query preserving watermarking based on alphanumeric data. For watermark embedding the presented scheme retains the semantic meaning of text data and securely adjusts the case of text data. The aim of the above techniques is to protect the copyright of databases, however, some studies focused on the verification of the authenticity of databases.

First pieces research in fragile database watermarking introduced distortions in original database regardless of the data constraints and may consider tuples sorting as malicious attacks. Guo, *et al.*, [7] securely partition database into groups and embed watermark in attributes At least two LSB, certainly embedding in LSB is sensible to modifications but easily violate data usability constraints. In [8], the embedding is also LSB based approach. Iqbal, *et al.*, [18] partition the database in 3 groups; they

generated and embedded in LSB the watermark bits. To overcome the distortion made to the databases during the embedding process, Yingjiu, *et al.*, [19] partition the database into disjoint groups and embed the watermark by reordering database tuples. Their technique can localize the tampering and doesn't modify any attribute values from the database. In [20], the watermarking scheme for protecting the integrity of database is presented. The technique uses database indexes for watermarking and doesn't change any attribute values. Moreover, it detects malicious attacks (insertion, deletion, modification, and displacement attacks) with high probability. Battacharya, *et al.*, [10] presented a virtual partition based fragile watermarking distortion free approach called partition abstraction, an abstract representation of the database concrete partitions is presented in such a way that any modification in the concrete domain is reflected in its abstract counterpart. An image of partition is generated and considered as partition watermark that serves as ownership proof as well as tamper detection.

3. Approach Overview

The Table 1 shows the notations and parameters used in this paper. We used the UTC (Coordinated Universal Time) date time stamp to generate the watermark bits b_i. Our fragile watermarking embedding algorithm takes a secret key K_S, and a watermark W as input to modify the database R into the watermarked database R_W. The field is modified by considering the predefined attributes usability constraints bound by the set Δ. These constraints control the amount of modification that a field can allow and we will discuss them in the next section.

The watermark encoding steps can be summarized as it follows:

3.1 Preparation Phase of Data Usability Constraints (Preprocessing)

Data values A_i in each attribute (column) of the data set are classified in different defined ranges according to their sensibility to authorize modifications.

3.2 Watermark Bits Generation Phase

The UTC date time is used to generate the watermark bits used in our encoding process.

3.3 Data Partitioning Phase

A secret key K_S and a cryptographic hash function (MD5) are used to partition the database R into different partitions having the same number of attributes and columns.

3.4 Attribute Selection Watermarking

To reduce the amount of database distortion, we select a single attribute in each group to embed the watermark.

3.5 Watermark Embedding

The fragile watermarking function embeds a single bit from watermark W to the selected attribute of each partition by verifying the data usability constraints Δ. The partition alteration is fulfilled by solving a constrained optimization problem. The constrained optimization problem solutions bound in τ are used to compute the optimal decoding threshold τ^*. The

computed watermarked data set is delivered to the intended recipient and might be subject to malicious attacks with the aim to modify the database without removing the watermark W .

The watermarking decoding is the process of extracting the watermark bits b_i from the watermarked data set R_W . It takes as input the following secret parameters known only by the database owner: secret key K_S , the optimal decoding threshold τ^* , the watermark length l , and the number of partition ν . But the number of partition ν is not a requirement for watermark decoding. The decoding watermarking algorithm does not use the original database R and the original watermark W to extract b_i and it is summarized as follows:

3.6 Data Partitioning

The data partitioning used is identical to the data partitioning technique used in the watermarking encoding phase.

3.7 Attribute Selection Watermarking

The marked attributes are identified using the same technique used in the embedding process.

3.8 Watermark Decoding

The computed optimal decoding threshold τ^* from the encoding phase is used to extract the watermark bits. The proposed decoding algorithm is blind.

4. Proposed Technique

4.1 Data Usability Constraints Preparation Phase (Preprocessing)

Let $R(P_k, A_0, \ldots, A_{\gamma-1})$ be the database to be protected. In our scheme, the modification will not affect the primary key attribute P_k but only some values of the γ numerical attributes are modified.

In the database to be protected, data values from the same attribute can be classified in different interval so that each data value can belong to an interval set as showed in the following example. To define the data usability constraints for student score attribute, each score should belong to one of the following groups and any member of a group should not be a part of another group after watermarking embedding process: Distinction “80-100%”, Very good “70-79%”, Good “60-69”, Pass “50-59”, Fail “0-49”.

The data usability constraints noted $\Delta = \{\Delta_1, \ldots, \Delta_{\gamma-1}\}$ are set for each individual attribute by defining different ranges for each attribute. If an attribute value belongs to a range, the amount of authorized modification should be in its corresponding attribute range values. As result the usability constraint is preserved. Only the data value belonging to a range can be modified, but other data including primary key attributes will stay unchanged during the encoding process.

4.2 Watermark Bits Generation and Bit Index Selection for Encoding

The UTC (Coordinated Universal Time) time stamp is used to generate watermark bits. The alpha numeric UTC is first transformed into binary string b_i . Consider the watermark

with a bits length l ' $W = b_i....b_{l-1}$. In this case, the watermark bit is inserted many times in the database as discussed below.

The bit string b_i index to insert in S_j is selected as follows: $i = k \bmod l$ (with K the group index) and as result the partition $S_j^{'} \leftarrow S_j$

4.3 Data Partitioning

The proposed partitioning algorithm is an extension of Shebab, *et al.*, approach, it divides the database R composed of α tuples and γ attributes into v different groups. Algorithm 1 shows the parsing process used in this paper.

Input : Database R , Number of groups v , and Secret key K_S

Output: ν groups $(S_1, \ldots, S_v)$ of length γ each

1. **Begin**
2. **for** i = 1 잼? α **do**
3. $h_i^r = Hash(k_S \| r_i.P \| k_S)$ // i^{th} row primary key hash
4. $j = h_i \bmod v$ // group index
5. insert r_i into S_j
6. Sort all tuples in S_j according to the increasing order of their primary key hash
7. **end for**
8. **return** $(S_1, .., S_v)$
9. e**nd.**

Algorithm 1: Database partitioning

A cryptographic hash function (MD5) is computed for each tuple and a number of few tuples r_i equal to γ are inserted logically into partitions $G_{j(1 \le j \le \gamma)}$ with $G_i \neq G_j \quad for \quad i \neq j$

$$j = \text{Hash}(k \| r_i.P \| k) \bmod v \tag{3}$$

As a result, the database R is partitioned into v different groups $\{G_1, ..., G_v\}$.

In the case where $\alpha \bmod \gamma \neq 0$, we simply securely insert records to complete the last group in order to get a square matrix and we make sure there is no identical tuple in the same group. Note that the added records will be deleted after the watermark insertion

The major advantage of such partitioning method is that many properties of square matrix can be applied to check the group's data integrity.

4.4 Attribute Selection for Watermarking

We consider each group as a square matrix, let D_j be the determinant of the j^{th} group. To complete the candidate attribute value to be marked, we compute k such that $k = \lfloor D_j \bmod \gamma \rfloor$ (γ represents the number of database attributes candidate for marking) and the attribute value $a_{k,k}$ is selected from the j^{th} partition to encode watermark bit b_i.

4.5 Watermark Embedding

In this part, the watermarking embedding algorithm reported in algorithm 2 is described. The algorithm first partitions the database R into v different partitions.

A single watermark bit b_i position is computed from watermark W and inserted in each partition S_j of the database R to obtain the altered partitions $S_j^{'}$ which are collected to get the watermarked database R_W .

Input: Database R , Watermark $W = \{b_i \ldots . b_{l-1}\}$, Secret Key K, Number of partition ν , Set of data usability constraints $\Delta_i = \{\Delta_1, \ldots, \Delta_{\gamma-1}\}$.

Output: Watermarked data set R_w , Optimal decoding threshold $\tau^* = \{\tau_0, \ldots, \tau_{\nu-1}\}$

1. **Begin**
2. **initialize** R_W , $\tau = \{\max_{S_j}, \min_{S_j}\}$
3. Data set partitioning into groups S_j // see algorithm 1
4. **for** each partition S_j
5. compute the determinant D_j of S_j // j^{th} determinant group
6. compute $k = \lfloor D_j \bmod \gamma \rfloor$ // index of attribute to be altered from the group
7. select $a_{k,k}$ from j^{th} group for alteration
7. compute b_i with $i = q \bmod l$ // (q the group index, l the watermark length)
8. **if** (b_i = = 1)
9. maximize D_j subject to Δ_i // insert b_i into S_j to get $S_j^{'}$ (matrix)
10. insert $D_j^{'}$ into $\max_{S_j}$ // computed determinant value
11. **else**
12. minimize D_j subject to Δ_i // b_i insertion into S_j to get $S_j^{'}$ (matrix)
13. insert $D_j^{'}$ computed value into $\min_{S_j}$ // computed determinant value
14. insert $S_i^{'}$ into R_W
15. compute optimal threshold τ^*
16. **end for**
17. **return** R_W , τ^*
18. **end.**

Algorithm 2: Watermark embedding

The index of the attribute $a_{k,k}$ to be altered in individual partition is computed using the formula (4), D_j represents the determinant value of j^{th} partition, γ the number partition attribute, and $\lfloor \ \rfloor$ the mathematical floor function.

$$k = \lfloor D_j \bmod \gamma \rfloor \qquad (4)$$

The watermark can be any selected l -bits binary string given by the data owner. In our experiment we use the UTC (Coordinated Universal Time) data-time stamp as watermark to be embedded in the database. The single bit b_i selection from watermark $W = \{b_i ....b_{l-1}\}$ to be embedded into S_j is computed using the formula (5), l is the length of watermark, and k the partition index.

$$i = k \bmod l \tag{5}$$

The bit encoding in partition is performed as follows: the encoding function first computes the partition determinant values, select the b_i from watermark to be embedded $a_{k,k}$ into S_j if $b_i = 1$, the determinant value of S_j is computed for different value of attribute subject to $a_{k,k}$ usability constraints. The maximum value of computed determinant is selected and logged respectively into $\max_{S_j}$ of threshold τ . The partition S_j modifies the attribute value $a_{k,k}$ to $a'_{k,k}$ and as result the partition S_j will change to altered partition S'_j. If the inserted watermark bit $b_i = 0$, the process remains the same only the maximum determinant value is change to the minimum determinant value and the determinant value will be logged into set $\min_{S_j}$ of the threshold τ .

The optimal decoding threshold τ^* can be computed using formula (6)

$$\tau_j^* = average\left(D_j, D'_j\right) \tag{6}$$

The value D_j represents the j^{th} partition determinant value and D'_j the j^{th} altered or watermarked partition determinant value.

4.6 Watermark Decoding

The watermark decoding algorithm consists of extracting the embedded watermark from the watermarked database.

The watermark decoding process is described in Algorithm 3. It takes as input the watermarked database R_W, the watermark length l, and the secret parameters k, ν, and τ^*. The watermark decoding algorithm first initializes the watermark length l to receive a set of l extracted bits from R_W partitions S'_j. Afterwards, R_W is partitioned in ν partitions using the technique used in the watermark embedding phase. As watermark is embedded several times, it should also be extracted several times. The number of partitions is the same as the number of computed optimal threshold and they are in the same order. For the watermark bit detection, there are 2 conditions for each partition: first the computed R_W partition S'_j determinant D_j^w should belong to $\left\langle D_j, D'_j \right\rangle$, second the partition determinant is computed and compared to its corresponding optimal decoding threshold τ_j^*. The watermark bit extraction is showed in algorithm 3. From line 7 to line 14, if the partition determinant D_j^w is greater than the optimal threshold τ_j^*, the extracted bit b_i is set as 1. If the partition determinant D_j^w is less than the optimal threshold τ_j^*, the extracted bit b_i is set as 0.

If D_j^w and τ_j^* are the same, the extracted bit b_i is set as T . Otherwise, the extracted bit b_i is set as F an erasure. Note that the decoding is independent of data usability constraints.

The watermark W_i is extracted many times (ν / l times) and it will be compared with the original watermark W for integrity purpose.

Input: Watermarked database R_W , K, ν , τ^* and Watermark length l
Output: Detected Watermark W_i
1. Begin
2. set $W_i[0,\ldots,l-1] \leftarrow 0$
3. R_W partitioning into groups $S_j^{'}$ // see algorithm 1
4. for each partition $S_j^{'}$
5. for $j = 0,\ldots, l-1$
6. compute the determinant D_j^w of $S_j^{'}$ // j^{th} determinant group
7. if $D_j^w \in \left\langle D_j, D_j^{'} \right\rangle$ and $D_j^w \succ \tau_j^*$
8. $W_i(temp) \leftarrow 1$
9. else if $D_j^w \in \left\langle D_j, D_j^{'} \right\rangle$ and $D_j^w \prec \tau_j^*$
10. $W_i(temp) \leftarrow 0$
11. else if $D_j^w \in \left\langle D_j, D_j^{'} \right\rangle$ and $D_j^w = \tau_j^*$
12. $W_i(temp) \leftarrow T$
13. else
14. $W_i(temp) \leftarrow F$
15. end if
16. end for
17. $W_i \leftarrow \{b_0^{'},\ldots, b_{l-1}^{'}\}$
18. end for
19. return W_i

Algorithm 3: Watermark detection

5. Effectiveness and Performance

In order to evaluate our approach, the effectiveness and performance of our algorithm to insert and detect the watermark from database are presented. For experiment purpose, we ran our algorithm on a subset of the Forest Cover Type [13] dataset, a real life database containing 581,012 tuples, each constituted with 10 integers attributes, 1 categorical attribute, and 44 Boolean attributes. We added a primary key attribute to the dataset to securely partition the dataset into ν groups or partitions as showed in algorithm 1. Moreover, the dataset 10 first integer attributes are used for watermark insertion and detection.

Experiments have been run on Duo Core Computer, 2GHz with 2 GB of memory and using SQL 2008 and Java/JDBC connectivity on Windows XP Professional.

Our watermarking embedding and detection algorithm selects one numerical attribute from each group to embed a single watermark bit. In the experiment, the watermark embedding and detection time is measured.

The imperceptibility of our approach is guaranteed by minimizing the database alteration and by controlling the modification of the attribute value from the partition subject to the attribute value usability constraints. Note that not always the maximum or the minimum value of data usability constraints values is chosen to alter the partition. As result, even if an attacker knows the data usability constraints, he cannot recognize the alerted data from the partition. We also conduct series of experiments to show the efficiency of our proposed watermark detection algorithm ability to detect malicious modifications made to the dataset. The four different attacks covered in this paper are: tuples insertion attack, tuples deletion attack, tuples modification attack, and multifaceted attack.

The attribute value usability constraints are set as preconditions. To simplify the experiment, the same data ranges are set for all the 10 candidate numerical attributes for watermark embedding and for experiment purpose, two ranges are used and constituted by a sequence set of 50 consecutive integers and a sequence set of 5 consecutive integers such that each attribute value belongs to one set. The proceeding time depends largely on the defined size of attributes range used for usability constraints whether it is tight or large. The time for tight range usability constraints is much smaller than for large defined range usability constraints.

To check the integrity of the database, each extracted l bits watermark is compared respectively to its position of original watermark l bits. If the two watermarks differ, the non integrity of the database is justified. Furthermore the non match bit b_i indicated the index of altered group.

5.1. Insertion Attack

In this attack, some tuples are randomly inserted progressively in the database with the aim to do not disturb the watermark. The insertion rate is set from 1% up to 50% of the database tuples. The attacker is faced to two problems: first if the inserted tuple is greater than 10 (the number of database attributes to be marked), the number of partitions will increase. Keeping the number of groups is not a requirement but it easily proves the integrity of the database at grouping stage by having different group number, and second the tuples insertion will greatly disturb the database groups. Our pilot study reveals that even if one tuple is inserted in the database, the non integrity of database is proved at 100%. Our technique is highly resilient against insertion attack as shown in Figure 1(a).

5.2. Deletion Attack

We next tested our algorithm against tuples deletion with different rate from 1% up to 50% of the original dataset. When a certain tuples deletion threshold is reached, the number of database groups will decrease and we generated different groups from the watermark embedding process and the tampering is detected at group level. As the watermark detection used determinant value which is sensible to a small change, accordingly the watermark will not be correctly detected from the database even if one tuple is deleted. The experiment results displayed in Figure 1(b) shows that our detection algorithm is highly resilient to tuple deletion attack. This attack inserts great distortion on extracted watermark after a tuples deletion consequently, the tampering is detecting at 100%.

5.3 Alteration Attack

We first randomly modified a single field from the database and we detected the watermark bit as "F" (an erasure). Afterwards we randomly modified the data values of database tuples progressively with different rates. The modification does not affect the primary key. The Figure 1(c) shows the results of watermark verification for attribute values modification. The watermark is not correctly detected from all modified groups. The attacker does not have access to the original database; he may violate the data value usability constraints. Even if the attacker knows the data constraints, the modification affects the group determinant. Therefore the watermark bit will not be extracted correctly. Our approach is resilient to alteration attack accordingly and the tampering is detected at 100%.

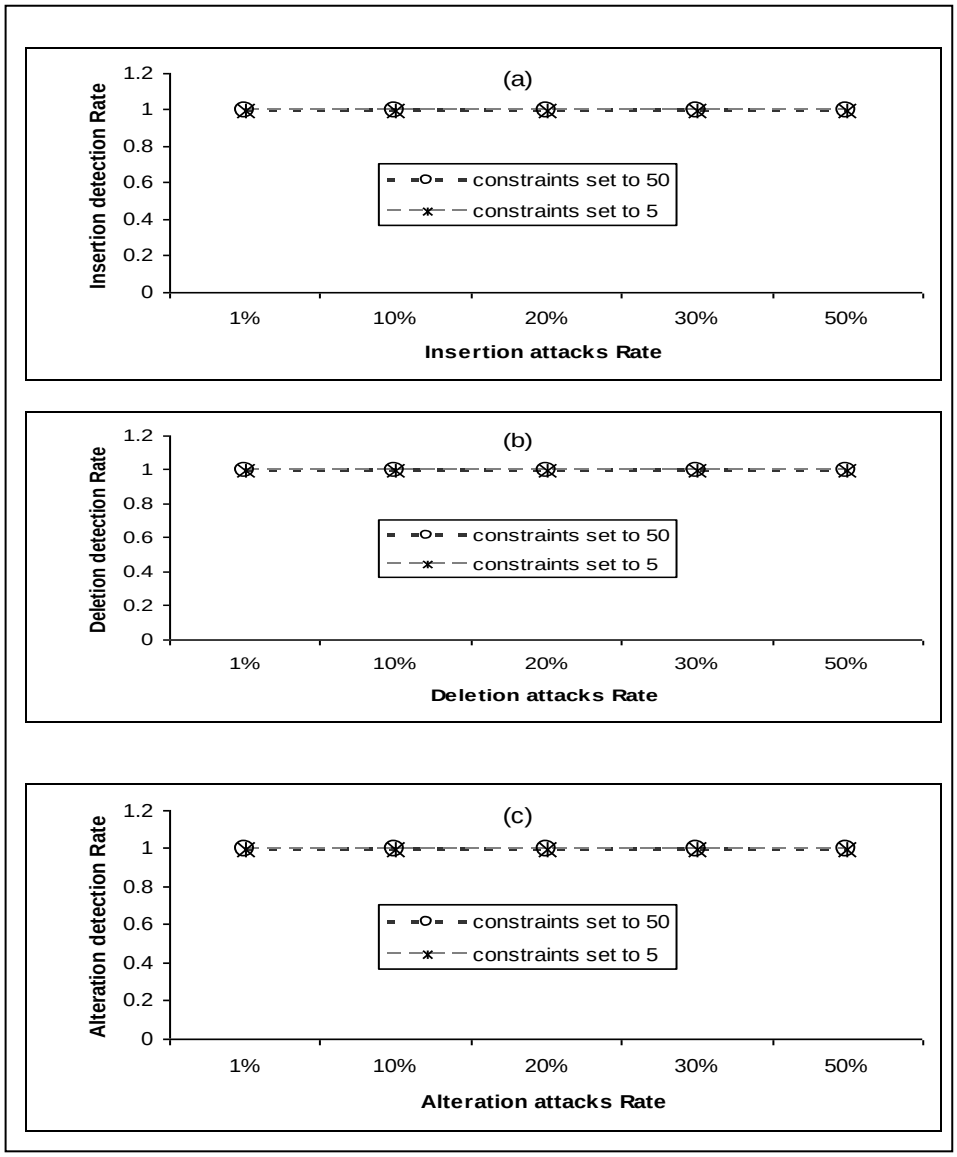

Figure 1. Resilience to Malicious Attacks

5.4 Multifaceted Attack

This attack is taken by a sophisticated attacker. This is the most challenging attack to test our algorithm. An attacker can interchange deletion, insertion and alteration in the database with the aim to modify the database without altering the watermark. The determinant value changed inevitably even after a small data value modification. As the attacker has no access to the secret key, he does not have any knowledge on the partition and the modified attribute cannot satisfy his desire. We have simulated attacks by inserting, removing the same tuples number and changed some field values with different rate. Our study reveals that the watermarking detection algorithm detects modifications made to the database at 100% whether it is small or high.

6. Discussion

6.1. Security

Our approach is secure against any malicious attacks having the aim to modify the database without altering the watermark and the data attributes usability constraints. We used a secret key only known to the database owner to securely partition the database into different square matrices and to securely sort tuples in the partitions. Furthermore, the partition index is used to embed the watermark bit. The watermarking embedding and detection algorithm used the mathematical determinant computation which is sensible to unauthorized modification made to the matrix element. As the attacker may not be able to generate database partitions, it is computationally hard even impossible, to have two square matrices having same order and same determinant value subject to data owner predefined usability constraints.

6.2. Embedding Distortion Amount Made to Database

Our technique minimizes the amount of alteration made to the database; at most the watermarking embedding algorithm modifies a single attribute value depending on if the attribute value allows distortion. Our scheme preserves field usability constraints by defining different ranges for each database attribute for watermark bit encoding purpose.

6.3. Blind Detection

One requirement of watermarking technique is blindness. The original database and the watermark are not used in our approach in watermark detection process, only the length of watermark is required.

6.4. Integrity Checking and Alteration Localization

The watermark bit is extracted consecutively from each partition using algorithm 4, each watermark length l is compared to the original watermark for integrity verification purpose. If the original and the extracted watermark are the same, the database has not been modified else the database has been tampered. The non match watermark bits position localizes the modified group.

6.5. Multi-bit Insertion

Our scheme is based on single bit insertion at group level but can support multi bit insertion. The same steps used in single bit embedding can be used for embedding multi bits. At group level, to embed a bit bi the embedding algorithm embeds bi in each group element

by considering their usability constraints. The watermark embedding can be formulated as constrained optimization problem. According to watermark bit value, the embedding algorithm modifies the partition attribute values by computing the maximum or the minimum determinant value. Intelligent algorithm such as Particle Swarm Optimization (PSO) [20] may be suitable for solving such problems.

7. Conclusion

In this paper, we have considered the embedding watermark bits problem as an optimization problem. We have partitioned a database into groups of square matrices and embedded a watermark bit into its selected diagonal value. The maximum or minimum group determinant is used to solve the problem. Furthermore, we have presented an efficient way to compute the optimal watermark decoding threshold. We have implemented our algorithm and experimental results showed that the proposed watermarking technique is reliable on detecting any malicious alteration target on the database.

Our scheme allows data values modification only on group diagonal values. In future works, we are planning to carry out more experiments to show the reliability our technique and work on semi fragile watermarking by extending the field value modification into the entire database. Besides this, we are also planning to investigate on the attribute values constraints so as to allow some modifications without altering the data usability constraints. We are also planning to extend our approach to non numeric values.

Acknowledgements

This project is sponsored by “the Scientific Research Foundation for the Overseas Chinese Scholars”.

References

[1] F. Guo, J. Wang, Z. Zhang, X. Ye and D. Li, “An improved algorithm to watermark numeric relational data”. Proceedings of the 6th International Workshop on Information Security applications, vol. 3786, **(2005),** pp. 138-149, Springer, Jeju Island, Republic of Korea.

[2] J. Lafaye, D. Gross-Amblard, C. Constantin and M. Guerrouani, “WATERMILL: an optimized fingerprinting system for databases under constraints”, IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 4, **(2008**), pp. 532-546.

[3] M. Kamran, A. Suhail and M. Farooq, “A Robust, Distortion Minimizing Technique for Watermarking Relational Databases Using Once-for-All Usability Constraints”, IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 12 (**2013**), pp. 2294-2707.

[4] M. Kamran and M. Farook, “Information Preserving Watermarking Scheme for Right Protection of EMR Systems”, IEEE Transactions on Knowledge and Data Engineering, vol. 24, no. 11, (**2012**). pp. 1950 – 1962.

[5] S. Liu, S. Wang, R. H. Deng and W. Shao. “A block oriented fingerprinting scheme in relational database”, Proceedings of the 7th International Conference in Information Security and Cryptology, **(2004)**, pp. 455–466, Springer-Verlag Berlin.

[6] Y. Li, V. Swarup and S. Jajodia, “Fingerprinting Relational Databases: Schemes and Specialties”, IEEE Transactions on Dependable and Secure Computing, vol. 2, no. 1, **(2005)**, pp. 34-45.

[7] H. Guo, Y. Li, A. Liu, and S. Jajodia, “A fragile watermarking scheme for detecting malicious modifications of database relations”. Information Sciences, vol. 176, no. 10, **(2006)**, pp. 1350–1378.

[8] H. Khataeimaragheh and H. Rashidi, “A novel watermarking scheme for detecting and recovering distortions in database tables”, International Journal of Database Management Systems, vol. 2, no. 3, **(2010)**, pp.1-11.

[9] A. Hamadou, X. Sun, S. A. Shah and L. Gao, “A Weight-based Semi-Fragile Watermarking Scheme for Integrity Verification of Relational Data”, International Journal of Digital Content Technology and its Applications, vol.5, no. 8, **(2011)**, pp. 148-157.

[10] S. Bhattacharya and A. Cortesi, “Distortion-free Authentication Watermarking”, Software and Data Technologies Communications in Computer and Information Science, Springer Berlin Heidelberg, vol. 170, no. 21, **(2013)**, pp. 205–219.

[11] S. A. Shah, X. Sun and A. Hamadou, "Query Preserving Relational Database Watermarking", Informatica, vol. 35, **(2011)**, pp. 391–396.
[12] D. Hanyurwimfura, Y. Liu and Z. Liu, "Text format based relational database watermarking for non-numeric data", Proc of IEEE International Conference on Computer Design and Applications (ICCDA), (**2010**) June 25-27, Qinhuangdao, China.
[13] http://archive.ics.uci.edu/ml/datasets/Covertype.
[14] K. H. Rosen, "Discrete Mathematics and its applications", sixth edition. McGraw-Hill Education (Asia), **(2007)**, pp. 247-256.
[15] R. Agrawal, P. J. Haas, and J. Kiernan, "Watermarking relational data: framework, algorithms and analysis", The International Journal on Very Large Data Bases (VLDB), vol. 12, no. 2, **(2003)**, pp. 157-169.
[16] M. Shehab, E. Bertino and A. Ghafoor, "Watermarking Relational Databases Using Optimization-Based Techniques", IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 1, **(2008)**, pp. 116 – 129.
[17] R. Schachtner, G. Pöppel, A. M. Tomé and E. W. Lang, "Minimum Determinant Constraint for Non-negative Matrix Factorization", Independent Component Analysis, Springer verlag, **(2009)**, pp. 106-113.
[18] S. Iqbal, A. Rauf, S. Mahfooz, S. Khusro and S. H. Shah, "Self-constructing fragile watermark algorithm for relational database integrity proof", World Applied Sciences Journal, vol. 19, no. 9, **(2012)**, pp. 1273-1277.
[19] Y. Li, H. Guo and S. Jajodia, "Tamper detection and localization for categorical data using fragile watermarks", Proceedings of the 4th ACM Workshop on Digital Rights Management, ACM Press, (**2004**) October, pp. 73-82, Washington DC, USA.
[20] I. Kamel, "A schema for protecting the integrity of databases", Computers & Security, vol. 28, no. 7, **(2009)**, pp. 698-709.
[21] K. Mahadevan and P. S. Kannan, "Comprehensive learning particle swarm optimization for reactive power dispatch", Applied Soft Computing, vol.10, no. 2, **(2010)**, pp. 641–652.

Authors

Camara Lancine, he is currently a PhD researcher at the College of Information Science and Engineering, Hunan University. His research interests include information security, and database application testing.

Junyi Li, he was born in 1970, PhD and associate professor at the College of Information Science and Engineering, Hunan University His main a research interest includes software engineering and database security.

Li Renfa, he was born in 1957. Professor and PhD supervisor at Hunan University, senior member of China Computer Federation (CCF). He has published over 60 scholarly papers in journals, book chapters and international conferences and acted as editor of a dozen books in recent years. His main research interests include CPS, embedded systems and wireless sensor networks. Prof. Renfa has been a chief investigator on research grants from different sources including Natural Science Foundation of China (NSFC) and others at state or provincial level.

Faustin Kagorora, he is currently a Master student at the College of Information Science and Engineering, Hunan University, China. His research includes Database Application Testing and Web Security Assessment.

Damien Hanyurwimfura, he received his Masters degree of Engineering in Computer Science and Technology from Hunan University in 2010. He is currently a PhD student at the College of Information Science and Engineering, Hunan University, China. He is also teaching at the College of Science and Technology, University of Rwanda, Rwanda. His current research interests include information security and data mining.