A Robust, Distortion Minimization Fingerprinting Technique for Relational Database

Namrata Gursale
Department of Computer Engineering
DYPSOET Lohegaon
Pune, India
e-mail:namrata.sgursale@gmail.com

Arti Mohanpurkar
Department of Computer Engineering
DYPSOET Lohegaon
Pune, India
e-mail:yasharti@gmail.com

***Abstract*— With** the widespread popularity of internet, the use of digital data has increased tremendously. In this context, it is essential to protect the ownership of the data and to find the guilty user. In this paper, a fingerprinting scheme is proposed to provide protection for relational database (RDB), which basically focuses on challenges like 1.Minimum distortion in database. 2. Usability constraints non-violation.3.Robustness against collusion attack. 4. Should not violate the requirement of blind decoding. The proposed fingerprinting scheme focuses on all these requirements.

Keywords- *distortion free; fingerprinting; tardos code; watermarking*

*****

I. INTRODUCTION *(HEADING 1)*

Fingerprinting has been used to protect valuable digital data such as documents, images, audio, video, databases etc. from illegal redistribution. Today, data mining tools are used to extract interesting patterns from relational databases which includes sharing of databases. In this context, some techniques like watermarking are used to provide ownership protection and Fingerprinting (also known as traitor tracing) to find out the source of data as well as unauthorized user/s (traitor/s) who redistribute the data.

In Fingerprinting, before distribution of data to user, it is required to first mark each numerical copy with unique identifier (imperceptible fingerprint) of user. If distributor finds the illegal copy of data on network then the user who might be responsible for creation of unauthorized copy is traced back. In our day to day life, there are, however, many applications, context of whose data represents an important asset, so the ownership protection and traitor identification must be carefully enforced. For example weather data, stock market data, medical, power consumption and scientific data etc.

Fingerprint embedding for relational data is made possible by considering user constraints which can tolerate a small amount of error [5]. Moreover, distortions in the original data are kept up to certain limits by introducing usability constraints, to preserve the knowledge contained in the database.

Embedding fingerprint to generate unique copies of digital document is a very natural process. The problem will not arise if buyer does not cheat but pirates may try to destroy fingerprint and redistribute illegally. So, to prevent and hence detect the fraud, unique mark is inserted in digital document at locations that are unknown to user [4].

The major contribution of the work presented in this paper is-

1. Fingerprinting the database while considering statistical usability constraints.
2. Preservation of knowledge i.e. minimum distortion in database.
3. Blind Decoding.
4. Finding the guilty user/s that is responsible for redistribution of unauthorised copy.

II. LITERATURE SURVEY

Most of research on fingerprinting is done on multimedia data. However, techniques used for multimedia data are not applicable as it is to relational data because it's properties as well as data operations are different. For example, Multimedia operations are like zooming and compression, while relational database (RDB) operations are tuple addition, deletion, and modification [12].

The watermarking scheme proposed by Agarwal et. al [10] is based on numeric attributes and marking is done at bit-level. The basic idea of this scheme is to ensure that some bit positions for some of the attributes of some of the tuples in the relation contain specific error values. This bit pattern constitutes the watermark.The parameter selection for watermarking is based on computing message authenticated code (MAC), where MAC is calculated using secret key and the tuple's primary key. Although LSB-based data hiding techniques are efficient, but shifting the LSB by only one position may lead to watermark loss without much damage to the data. This technique assumes unconstrained LSB manipulation during watermark embedding process.

Yingjiu Li [12] presented a technique for fingerprinting relational data by extending Agrawal et al watermarking scheme. Several measures for the robustness of fingerprinting scheme are defined, with solution suggested for collusion attacks. This scheme can be used for both watermarking and fingerprinting. But the usability of data is not considered.

While inserting fingerprint, JulienLafaye [3] presented watermarking/fingerprinting system for relational databases. It features a built-in declarative language to specify usability constraints. Two watermarking strategies: Integer Linear algorithm and collusion secure pairing algorithm using

 ISSN: 2321-8169
1737 – 1741

Tardos code for fingerprinting are defined. It also considers local and global constraints on data. But again watermark insertion is performed by modifying the LSB of numerical values.

In [6], M. Kamran et al developed a watermarking scheme that is able to meet the challenges such as robustness against different attacks, preserves the knowledge in the database, try to strike balance between the conflicting owner and buyer usability constraints and once for all usability constraints. The proposed algorithm embeds every bit of a multibit watermark in each selected row. As a result the proposed scheme achieves 100 percent decoding accuracy even if only one watermarked row is left in the database.

Consider an example [3], three fingerprints m_1= 0110, m_2=1000, m_3=1110 used for three users u_1, u_2 and u_3 respectively. User u_1 and u_2 compare their respective copies and modify it to build a copy where it differs. So, the fabricated fingerprint is 1110 which is actually the fingerprint for user u_3. So it can happen that user u_1 and u_2 actually collude to pirate the database but user u_3 is identified as victim after detection. To avoid accusing such innocent users collusion-secure codes have been designed for example Boneh and Shaw, Tardos code.

In [2], Boneh and Shaw presented a binary randomized code, which uses concatenation of partly randomized inner code with outer code. They have also proposed Marking Assumption where colluder cannot alter the marks at which their copies agree.

In [4], Tardos described a fully randomized binary fingerprinting code further tightening lower bound. This fingerprinting scheme is particularly known for its short code length. While using tardos code owner does not have to know the number of user in advance, new user can be added dynamically. And B. Skoric et al. [1], proposed a symmetric version of the original Tardos score.

III. SYSTEM ARCHITECTURE

Figure 1 shows a block diagram summarizing the main components of the fingerprinting system model used. Tardos method is used to generate the fingerprint code. A robust fingerprint algorithm is used to embed fingerprint into original database. The algorithm takes a secret key (K_s) and fingerprint $\boldsymbol{f_c}$as input and converts numeric database $\mathbf{N_{DB}}$ into fingerprinted version $\boldsymbol{f_{DB}}$ with usability constraints U_c. In this technique, U_c is defined only once for database used all possible type of applications [6].

Fingerprint Encoding:

Tardos Code Generation: Fingerprint is created by using Tardos Code ($\boldsymbol{f_c}$). These bits are given as input to Fingerprint embedding process.

Data Partitioning: The Database $\mathbf{N_{DB}}$ is partitioned into m non-overlapping partitions by using secret key (K_s) concatenated with cryptographic secure hash function H().

Subset Selection: In this process, few tuples are selected for fingerprinting to minimize the distortions.

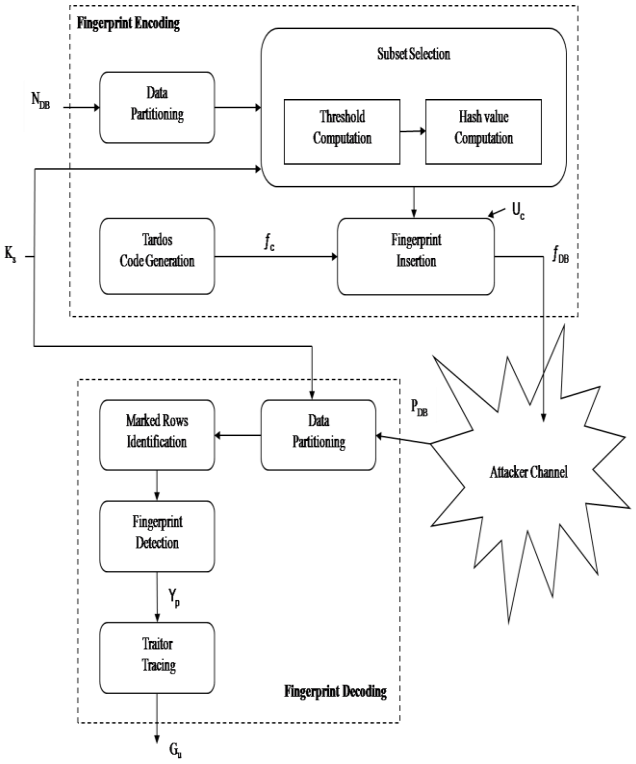

Figure 1: Proposed System Architecture

Fingerprint Insertion: Fingerprint bits are embedded in the selected tuple of each partition by using fingerprinting function.

Fingerprint Decoding:

It is the process of capturing the embedded fingerprint codeword from pirated copy (P_{DB}) using secret key K_s, correction factor C_f and fingerprint decoding threshold γ. After extracting embedded fingerprint Tardos decoding algorithm is used to find out guilty user (G_u).

Data Partitioning: Same data partitioning algorithm is used to partition the data as used in the fingerprint encoding phase.

Marked Rows Identification: Fingerprinted rows are identified by the same procedure used while inserting fingerprint in encoding phase.

Fingerprint Detection: As of mentioned decoding does not violate the requirement 'blind decoding'. The decoding algorithm decodes the inserted fingerprint without considering usability constraints. Only modifications done at the time of embedding process are taken into consideration.

Traitor Tracing: Captured fingerprints are the input to the traitor tracing process. Using captured fingerprints,one can detect guilty user by comparing captured fingerprint to each buyer's fingerprint.

IV. PROPOSED ALGORITHM

A. Data Partitioning

The dataset N_{DB} is database relation with scheme N_{DB} = (P_k, A_0, …… ,A_{n-1}), where $P_k$is the primary key and

$A_0,\ldots,A_{n-1}$ are n attributes. The partitioning algorithm divides the data set into m non-overlapping partition namely $\{P_0,\ldots,P_{m-1}\}$ such that for any two partition $P_i \cap P_j=\{\}$.The data partitioning algorithm proposed in [6][9] has been used to partition data set into groups. For each tuple $t \in N_{DB}$, the data partitioning algorithm computes MAC to assign tuples to the partition using hash function H as-

$$\text{Partition}(r) = H(K_s \,||\, H(t.P_k \,||\, K_s)) \bmod m$$

where, $r.P_k$is primary key of tuple r, H() is secure hash function and || is concatenation operator.

Algorithm 1: Get_partitions.
Input: Data set N_{DB}, Secret key K_s, Number of partition m.
Output: Data Partition $P_0,\ldots,P_{m-1}$

1. for each tuple r $\in N_{DB.}$
2. Partition(r) = $H(K_s \,||\, H(t.P_k \,||\, K_s)) \bmod m$
3. insert r into $P_{Partition(r)}$
4. end for
5. return$P_0,\ldots ,P_m$

B. *Subset Selection*

Selection of data set for fingerprinting is done by two steps as proposed in [6]. These both steps reduce the number of tuples to be fingerprinted so distortions are minimized.

Threshold (T) computation:

$$T = f * \mu + \sigma \quad (1)$$

where, μ is the mean and σ is the standard deviation of values of attribute A in N_{DB}. f is confidence factor which is kept secret. Threshold is calculated for each attribute. A tuple is selected for marking if any of its attribute has a value above its corresponding threshold computed. And union of selected tuples are taken into consideration

Algorithm 2: Get_DataSelection_Threshold:
Input: Data Partition $P_0,\ldots,P_{m-1}$
Output: Data Set N_{DB1}

1. for i = 0 to m-1 **do**
2. for each A $\in P_i$**do**
3. Compute μ and σ on A
4. Calculate T using (1)
5. end for
6. end for
7. return $N_{DB1} \leftarrow R_{>T}$

Hash value computation:

The data set N_{DB1} is used to select tuples with even hash values. The hash function MD5 is applied on the selected data set N_{DB1}. And form the data set N_{DB2}.

Algorithm 3: Get_EvenHashValue_Dataset
Input: Data set N_{DB}, K_s
Output:N_{DB2}

1. for each r $\in$ N_{DB2} do
2. Even_value(r) = $H(K_s \,||\, r.P_k) \bmod 2$
3. if Even_value(r) = = 0 then
4. insert r into N_{DB2}
5. else
6. Tuple not considered for fingerprinting
7. end if
8. end for
9. return N_{DB2}

C. *Generation of Fingerprint Bits*

Fingerprint is generated by using Trados Fingerprinting code. Fingerprinting traces the guilty user(s) who redistribute(s) unauthorized data. The fingerprints should be inserted such that its location should not be revealed to the traitor. A well-known attack called Collusion Attack, where more than two pirates compare their copies and build their own as they want is important to be handled.

For example [5], Consider three fingerprints m_1 = 0110, m_2 = 1000, m_3 =1110 used for three users u_1,u_2 andu_3 respectively. User u_1 and u_2 compare their respective copies and modify it to build a copy where it differs. So, the fabricated fingerprint is 1110 which is actually the fingerprint for user u_3. So it can happen that useru_1 and u_2 actually collude to pirate the database but useru_3 is identified as victim after detection. To avoid accusing such innocent users collusion-secure codes have been designed like Boneh and shaw scheme code[7],Tardos Scheme[2]. Out of these Tardos scheme is better because [5].

1. Codeword has small length as compared to other Schemes.
2. Number of users need not be predefined.
3. Implementation is simple and efficient.

Following are the steps to generate Tardos Fingerprinting code: [10] Two Phase Process:

1. First distributor choose the codeword's distribution.
2. Fingerprinting matrix B_T is built. The rows of the matrix $B_T$are the codeword's embedded into the copies that the users receive.

Phase 1:

In this phase of the code construction the distributor picks a random variable. The distributor does it in the following way:

1. The distributor picks some suitable parameter $t_p$where t_p=1/(300c)
2. The distributor computes the parameter
3. For all $1 \le i \le N$ the distributor picks $r_i$uniformly at random from the interval
4. The distributor puts $p_i = \sin^2 r_i$, where $1 \le i \le N$ and thus $t \le p_i \le 1 - t_p$.

The distribution is therefore defined as follows

$$F(p) = \frac{1}{(\pi - 4t_p')\sqrt{p(1-p)}} \quad (2)$$

Phase 2:

Fingerprinting matrix B_T is built. The entries of B_T are such that $P[B_T(j,i)= 1] = p_i$ and $P[B_T(j,i)=0] = 1 - p_i$. Where, j denotes the number of user j = 1, …., n and i denotes the code length i = 1, …., *l*.

D. Fingerprint Insertion Algorithm

The tardos fingerprint generated bits $b_1, b_2, \ldots, b_l$, are inserted in each partition P_i. Fingerprint is inserted into dataset by using proposed algorithm in [6] but attribute are selected pseudo randomly with setting seed as concatenation of primary key, buyer id and secret key. Fingerprint embedding function (F^I) is used to convert dataset N_{DB} into fingerprinted database f_{DB}.

$$F^I : (N_{DB}, f_c) \rightarrow f_{DB} \quad (3)$$

Fingerprinting function F^I for row i and j^{th} column is-

$$F^I = \eta_{ij} + v_i \quad (4)$$

Where, parameter η_{ij} is one instance of modification performed on tuple i in jth column, computed according to b. If b is equal to one, then η_{ij}, subject to constraints U_s ,on data value v_{ij} of an attribute as –

$$\eta_{ij} = \tau\% \text{ of } v_{ij} \text{ with } \tau > 0 \quad (5)$$

and if b is equal to zero, then

$$\eta_{ij} = \tau\% \text{ of } v_{ij} \text{ with } \tau < 0 \quad (6)$$

where, τ is fixed for every fingerprinted tuple and is defined by owner. And used during fingerprint decoding. Usability constraints are defined in terms of mean and standard deviation. These two measures remain approximately the same before and after fingerprinting of data.

Algorithm 4: Insert_Fingerprint
Input: Data set N_{DB}, Data set N_{DB1}, f_c, Secret key K_s
Output: Fingerprinted Data set f_{DB}

1. $f_{DB} = N_{DB}$
2. N_{DB2} = Get_EvenHashValue_Dataset (N_{DB1},K_s)
3. for each tuple r in N_{DB}" do
4. select attribute pseudo randomly using secret key, primary key and buyer id
5. if b == 1 then
6. compute η from equation (5)
7. else
8. compute η from equation (6)
9. end if
10. $N_{DB(temp)} \leftarrow (\eta + N_{DB(temp)})$
11. end for
12. insert η into Δ
13. compute C_f
14. return f_{DB} and C_f

Data manipulation vector (Δ) keeps a record of transformation from N_{DB} to f_{DB} by adding parameter η in each encoding step. The outputted data set f_{DB} is then made available for the buyer. The value of correction factor must be less than the minimum absolute value of $\eta \in \Delta$ [6]. Algorithm 4 represents the steps to embed the fingerprint into dataset.

E. Fingerprint Detection

In [6], proposed blind decoding algorithm does not need the original data set, following algorithm has been used in our scheme to detect fingerprint bits. Fingerprint bits are extracted by using the parameters such as secret key, primary key, and secret key. Firstly, data partition algorithm is used to partition the data set f_{DB} into logical groups by giving input f_{DB}, Ks and m. After that f_{DB1},f_{DB2} is then computed as done in encoding phase. In the next step, depending upon Δ and Value, bit zero and one are decoded.

The parameter value can be calculated using relation-

$$\text{Value} = = \tau\% \text{ of } v'_{ij} \quad (7)$$

Where, v'_{ij} is value of j^{th} attribute in i^{th} tuple. The steps of fingerprint decoding are shown in algorithm 5.

Algorithm 5: Detect_Fingerprint
Input: Fingerprinted data set f_{DB}, K_s,m,C_f
Output: captured fingerprints $Y_p=\{Y_1,Y_2, \ldots ,Y_n\}$

1. $Pw_0, \ldots\ldots, Pw_{m-1}$ = Get_partitions(f_{DB}, K_s, m)
2. for each partition Pw_i do
3. f_{DB1} = Get_DataSelection_Threshold()
4. f_{DB2} = Get_EvenHashValue_Dataset()
5. for each tuple r in f_{DB2}**do**
6. select attribute pseudo randomly using secret key, primary key and buyer id
7. calculate value
8. if $\partial > Value$ then
9. decoded bit one
10. else
11. decoded bit zero
12. end if
13. end for
14. Apply majority voting for each decoded fingerprint
15. return Y_p

F. Traitor Tracing

In this process, the captured fingerprints for each user are compared with tardos fingerprint created code for each user. If captured fingerprint is matched with user fingerprint then that user is traced as a guilty user. Suppose, user1 is guilty user, In such situation according to decoding algorithm firstly for all user, inserted fingerprints are calculated that are $Y_p= \{Y_1,Y_2, \ldots ,Y_n\}$ and after that these captured fingerprint are checked one by one with already stored tardos fingerprint of each user $B_{IDS} = \{B_T(1, i), B_T(2, i),\ldots, B_T(n, i)\}$ where, j=1,..,n. Here, it is matched with user1. And it is traced as guilty.

V. Experiment and Result

In this section, results of our experiments are reported. Selected database is of 205 tuples. The number of partition m =20 and τ ±0.01 is used. Table 1 show the mean and standard deviation before and after fingerprinting database. Difference between before and after inserting fingerprint is minuscule. And it decodes the fingerprint correctly and trace the guilty

 ISSN: 2321-8169
1737 – 1741

TABLE I. EXPERIMENTAL RESULT

Sr.No.	Attributes	Before Insertion		After Insertion		Difference in %	
		Mean	*Std Dev*	*Mean*	*Std Dev*	*Mean*	*Std Dev*
1	Wheelbase	98.7565	6.0070	98.7784	6.0068	0.00021	0.000002
2	Length	136.4029	38.8718	136.4268	38.8729	0.00023	0.000011
3	Width	112.9045	45.9697	112.9238	45.9751	0.00019	0.000054
4	Height	98.1096	47.3610	98.1263	47.3674	0.00016	0.000064
5	Bore	79.14066	56.8665	79.1541	56.8750	0.00013	0.000085
6	Stroke	66.4825	59.1271	66.4938	59.1363	0.000113	0.000092
7	Compression Ratio	58.4339	58.2023	58.4439	58.2114	0.0001	0.000091

user accordingly. The database is used as automobile fields containing numeric fields.

I. CONCLUSION

In this paper, the proposed fingerprinting technique inserts the fingerprint bits subject to usability constraints. And results, minimum distortion in original data set as well as finds the guilty user who is responsible for illegal redistribution of data set. A logical extension of this research is to extend the technique on non-numeric strings data.

REFERENCES

[1] B.Skoric, S. Katzenbeisser, and M. Celik. "Symmetric Tardos fingerprinting codes for arbitrary alphabet sizes", Designs, Codes and Cryptography, vol 46, no. 2, pp.137-166, February 2008.

[2] D. Boneh and J. Shaw, "Collusion Secure Fingerprinting for Digital Data," IEEE Trans. Information Theory, vol. 44, no. 5, pp. 1897-1905, 1998.

[3] JulienLafaye, David Gross-Amblard, camellia Constantin, and MeryemGuerrouani,"WATERMILL : An Optimized Fingerprinting System forDatabases under Constraints", IEEE Transactions on Knowledge and Data Engineering,vol. 20,no. 4, April 2008.

[4] G. Tardos, "Optimal probabilistic fingerprint codes," in Proc. 35th Ann, ACM Symp. Theory of Computing, May 2003, vol . 55, no. 2, pp. 116-125[online].Available:http://portal.acm.org/citation.cfm?doid=1346331346335, ACM.

[5] Mohamed Shehab, Elisa Bertino, ArifGhafoor, "Watermarking Relational Databases UsingOptimization-Based Techniques", IEEE Transactions on Knowledge and Data Engineering,vol. 20,no. 1, 2008.

[6] M. Kamran, Sabah suhali, and MuddassarFarooq, "A Robust, Distortion Minimizing Technique for Watermarking relational Databases Using Once-for-all Usability Constraints", IEEE Transactions,vol.25,no. 12, 2013.

[7] M. Kamran and Muddasarfarooq, "An information-Preserving Watermarking Scheme for Rigth Protection of EMR Systems", IEEE Transaction on Knowledge and Data Engineering, vol. 24, no. 11, November 2011.

[8] Mohamed Shehab, Elisa Bertino, ArifGhafoor, "Watermarking Relational Databases UsingOptimization-Based Techniques", IEEE Transactions on Knowledge and Data Engineering,vol. 1020,no. 1, 2008.

[9] OdedBlayer and TamirTassa, "Improved versions of Tardos' _ngerprinting scheme", Des. Codes Cryptography, 48(1):79{103, 2008.

[10] R. Agrawal, P.J. Haas, and J. Kiernan, "Watermarking Relational Data: Framework, Algorithms and Analysis," The VLDB J., vol. 12,no. 2, pp. 157-169, 2003.

[11] T. U. Vladimirova, "Collusion Resistence of Digital Fingerprinting Schemes",Technische Master's Thesis, June 2006.

[12] Yingjiu Li, VipinSwarup, SushilJajodia, "Fingerprinting Relational Databases: Schemes And Specialties", IEEE Transactions,vol. 2, no. 1, 2005.