Research Journal of Information Technology, 2009
ISSN 1815-7432
© 2009 Academic Journals Inc.

A Robust Tamperproof Watermarking for Data Integrity in Relational Databases

V. Prasannakumari
Systems Analyst, Tek-Tools Software Pvt Ltd.,
2/36 Sripuram First Street, Royapettah, Chennai, India

Abstract: Watermarking for Databases is one of the fastest growing areas of research dealing with security and authentication issues for digital contents when they are shared or distributed to others. One of the security aspects is to authenticate tamperproof receipt of the database when it has been over any communication channel. My approach is to append a new attribute which will serve as a watermark containing checksum of all other attributes and an aggregate value for any one of the numeric attribute. Since, the values at none of the attributes are altered to accommodate watermarks, precision is preserved. The implementation proves the technique robust against common database attacks.

Key words: Database watermarking , tamperproof, robust, watermark embedding, watermark detection, database attacks

INTRODUCTION

Availability of easy access and high speed connections to internet has made businesses grow faster and smarter. Distribution of databases has become one of the vital activities in business. Buying/Selling of databases containing data that can be useful for business is also in practice. This database enables them to reach out the prospective domain of people or use the present data for further research and analysis. Tamper-proof transportation of databases is one of the recently faced issues in such cases. Organizations expect the databases to be the same without any tampering.

The Information Commissioners Office, United Kingdom (OUT-LAW News, 2006) has published a guidance note advising businesses on how to comply with the Data Protection Act when buying and selling databases containing customers confidential information. It says This good practice note will help businesses understand what they need to do to ensure that personal information on the databases is sufficiently protected. (OUT-LAW News, 2006, http://www.out-law.com/page-6826).

Digital watermarking techniques have emerged as an effective solution in addressing this problem. Originating from steganography which is the art and science of concealing the very existence of the secret message, digital watermarking provides proof of authentic data and tracking capability to illicit copying and distribution of digital data. Watermark describes information that can help in proving ownership and tamperproof. Secure embedding requires that the embedded watermark is imperceptible and should not be tampered easily. A blind detection process which does not require original database to compare against the received database is preferred to ensure integrity of data.

Early studies were to watermark relational databases for copyright protection, which assume that the relational data have enough redundancy and can tolerate some unnoticeable and negligible degradation in data precision caused by embedding watermarks. Their objective was to verify the copyright of the relational data and the ownership and the fragile watermarking scheme algorithm proposed by Li *et al.* (2004) was to detect and localize the tampered area of categorical data, which however, inevitably introduces permanent distortion to the cover data. Sensitive data in some applications cannot tolerate any permanent distortions and data's integrity needs to be authenticated. To meet this requirement, we propose a fragile watermarking which derives itself from the data that are stored in the database.

Agarwal and Kiernan (2002) used marker tuples which faced watermark synchronization problem. Zhang *et al.* (2004a, b, 2006) proposed methods to embed a watermark image into the database. Shehab *et al.* (2008) proposal was to use partitioning technique to overcome limitations with marker tuples. The watermark decoding was based on the threshold and was proved using majority voting technique.

In a robust watermarking scheme for ownership verification, an attacker attempts to take away the embedded watermark or make it undetectable while keeping the database relation useful. Thus, the design purpose is to make the embedded watermark robust against malicious attacks. In contrast, this scheme is a fragile watermarking scheme for tamper detection. In this kind of scheme, an attacker will try his/her best to make alterations to a database relation while keeping the embedded imperceptible watermarks untouched. The attack is successful if the database relation is altered while the embedded watermarks are still detectable. Thus, in my scheme, the embedded watermarks are designed to be derived from the data which makes them invalid so as to detect any modifications made to a database relation. The proposed technique is resilient to tuple deletion, alteration and insertion attacks.

Approach Overview

Figure 1 shows a framework of watermarking system model proposed. A data set D is transformed into a watermarked version D_w by applying a watermark encoding function. Watermarking here will not modify any data thus preserving the sensitivity of the data involved.

The watermark encoding can be summarized by the following steps:

- **Step E_1:** Data set D is partitioned using a user-chosen partition variable into *n* non-overlapping partitions $\{S_0, S_1 \ldots S_n\}$
- **Step E_2:** Watermark embedding: a virtual attribute V_A is added to the database whose value is composed from aggregate functions involving the data present in that particular partition. Watermark embedding will not alter values of the data rather introduces a checksum attribute to ensure tamperproof

Watermarked version of the database Dw is sent to the intended recipient. On the way, it may suffer unintentional attacks aimed at destroying the original data present. Watermark decoding is the process of extracting the embedded watermark using the watermarked data set D_W.

The watermark decoding is divided into four main steps:

- **Step D_1:** Data set partitioning: by using the data partitioning algorithm used in E1, the data partitions are generated

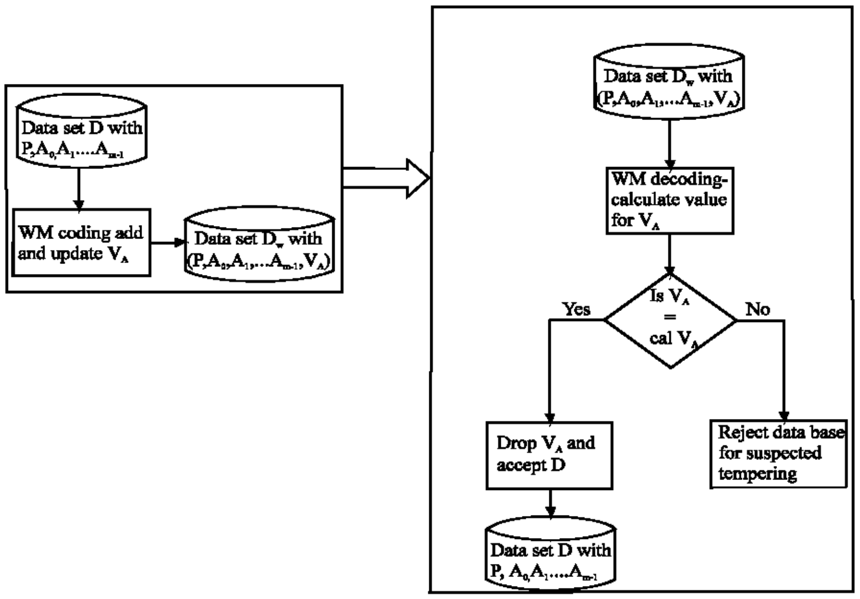
Data set D with P,A0,A1....Am-1
WM coding add and update VA
Data set Dw with (P,A0,A1,..Am-1,VA)
Data set Dw with (P,A0,A1,...Am-1,VA)
WM decoding-calculate value for VA
Is VA = cal VA
Yes
No
Drop VA and accept D
Reject data base for suspected tempering
Data set D with P, A0,A1....Am-1

Fig. 1: Framework of watermarking system model

- **Step D_2 :** Data at the virtual attribute added as watermark is checked against the new value by re-calculating it using the data present in the received dataset
- **Step D_3 :** The dataset is certified to be tamperproof if the calculated and the available values of the virtual attribute are same. Otherwise the database can be rejected for suspected tampering
- **Step D_4 :** The virtual attribute V_A added to hold the watermark is dropped

Data Partitioning

Here, we present the data partitioning algorithm that partitions the data set basedonapartitionattributeA_p. The data set D is a database relation with scheme $D(P,A_0, . . . ,A_{m-1})$, where P is the primary key attribute, $A_0, . . . ,A_{m-1}$ are attributes which are present and $|D|$ is the number of tuples in D. The data set D is to be partitioned into n non-overlapping partitions namely $\{S_0,S_1... S_n\}$ such that each partition S_i contains on average $\frac{|D|}{n}$ tuples from the data set D. Partitions do not overlap, that is, for any two partitions S_i and S_j such that $i \neq j$ we have $S_i \cap S_j = \{\}$. For a tuple r its partition assignment is given by:

$$\text{Partition}(r) = H(r.A_p) \bmod n$$

Using the property that secure hash functions generate uniformly distributed message digests this partitioning technique on average places $\frac{|D|}{n}$ tuples in each partition. Furthermore, an attacker cannot predict the tuples-to-partition assignment without the knowledge of the partition attribute chosen and the number of partitions m which are kept secret. Though keeping m secret is not a requirement, this can make it harder for him to regenerate the partitions.

This technique is not dependent on any attribute like primary key since this introduces a new virtual attribute to hold the checksum which will be dropped once decoded and certified for tamperproof receipt.

Watermark Embedding

Here, we describes steps in embedding watermark into the dataset. A new column is added to the existing schema introducing virtual attribute V_A Parity checks from each of the attribute is calculated and appended to V_A. V_A also holds a value from aggregate function of any of the numerical attribute for all the tuples in that partition.

Algorithm Encode

- Add a virtual attribute with default NULL values to the data set D
- For each partition r
- Calculate agg(r, A_i) where A_i is the chosen numeric attribute and assign to V_A of all tuple in that partition
- For each tuple t in D
- Calculate parity checksum for each attribute and concatenate in V_A

Watermark Detection

- Re-generate the partitions using the same procedure used for partitioning
- Recalculate the value for each V_A using the same calculations done while embedding
- Compare the calculated V_A with the V_A that is present in the table
- If they are same then certify the database to have been received without any tampering, reject otherwise
- Drop the attribute V_A from database. Accept the database

Attacks

Intentional attacks that are possible on a database could be any one of the following:

- Insert a new tuple into the dataset D
- Alter values of any of the attributes
- Delete some tuples from the dataset D

The proposed model is resilient to all of the above attacks. We assume a sample database as in Table 1 to be the simplest to the core for demonstration and to run the attacks on.

We choose attribute A_0 to be partition key and this could generate two partitions with 2 tuples in each for our sample database. Table after watermark embedding is shown in Table 2.

Table 1: Sample database for illustration

A_0	A_1	A_2	A_3
1	01/01/2009	Chennai	49.01
2	01/01/2009	Mumbai	48.75
1	01/03/2009	Chennai	48.65
2	01/03/2009	Mumbai	47.95

Table 2: Sample database after adding watermark

A_0	A_1	A_2	A_3	V_A
1	01/01/2009	Chennai	49.01	179797.66
2	01/01/2009	Mumbai	48.75	178696.70
1	01/03/2009	Chennai	48.65	199597.66
2	01/03/2009	Mumbai	47.95	198796.70

Data is not altered to embed watermark. Hence precision preserved

Table Watermark Embedding

Insertion Attack

Assume an attacker inserts an extra tuple into the existing set D. Table 3 shows the dataset after the attacker inserts a new record into D.

On detection process the parity check at the new tuples V_A will not match. In the event of matching by chance, the aggregate function for the partition will fail to match. Hence the value at existing V_A will differ from the calculated and hence the database can be rejected in suspect of tampering.

Alter Attack

If the attacker alters value of any of the columns, there is very less chance of it getting failed to differ from the calculated V_A as in Table 4.

On the other hand, if he attempts to alter the numeric attribute which is involved in the aggregate function then it would be revealed with more confidence (Table 4).

If he attempts to modify value at numeric attribute the result would be as in Table 5.

Deletion Attack

Since, the aggregate function is part of V_A, deletion of tuple will cause the aggregate function value to differ and hence the database can be suspected for tampering.

Overhead Metrics

Appending a new attribute V_A will occupy space and will increase the space required for the database. Since space constraints are no longer costlier due to the availability of huge storage systems at affordable costs, the overhead incurred is negligible.

For, the sample database with 12 attributes having 12800 tuples required, Size or Space before embedding watermark : 240 KB Size or Space after embedding watermark : 280 KB.

Table 3: Sample database in case of intentional insertion attack

A_0	A_1	A_2	A_3	V_A
1	01/01/2009	Chennai	49.01	179797.66
2	01/01/2009	Mumbai	48.75	178696.70
1	01/03/2009	Chennai	48.65	199597.66
2	01/03/2009	Mumbai	47.95	198796.70
1	01/02/2009	Chennai	48.95	179797.66*

*Since relevance is not revealed attacker would assign a random value for V_A

Table 4: Sample database in case of intentional alteration attack

A_0	A_1	A_2	A_3	V_A
1	01/01/2009	Mumbai	49.01	179797.66*
2	01/01/2009	Mumbai	48.75	178696.70
1	01/03/2009	Chennai	48.65	199597.66
2	01/03/2009	Mumbai	47.95	198796.70

calculated V_A on detection process will be 178797.66

Table 5: Sample database in case of intentional alteration attack

A_0	A_1	A_2	A_3	V_A
1	01/01/2009	Chennai	49.11	179797.66*
2	01/01/2009	Mumbai	48.75	178696.70
1	01/03/2009	Chennai	48.65	199597.66
2	01/03/2009	Mumbai	47.95	198796.70

Calculated V_A on detection process will be 178797.66

Table 6: Space requirement for the sample database

No. of attributes	Space consumed in KB	Size after watermarking in KB	Overhead space in KB	Overhead(%)
12	240	280	40	16.67
20	440	504	64	14.55
40	960	1084	124	12.92
80	2080	2324	244	11.73
120	3360	3724	364	10.83

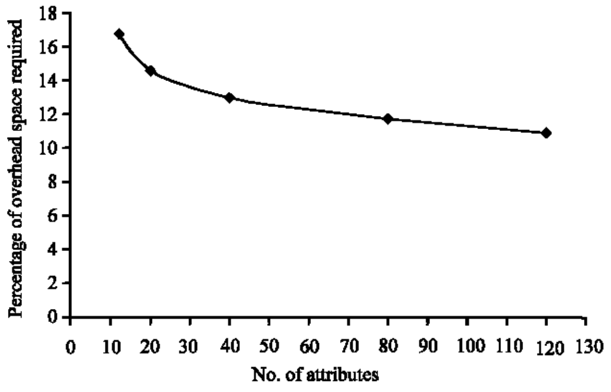

Fig. 2: Space overhead by introducing the new parameter to hold watermark

Overhead caused by the watermark = (280-240)/240*100 = 16.67. Table 6 shows the results for the overhead caused with different number of attributes.

Trade-off is around 17% increase in space consumed with 12 attributes. More the number of attributes, lesser would be the overhead space. Figure 2 shows the space overhead incurred by introducing the new parameter to hold watermark. From Fig. 2 it is evident that the space overhead deteriorates. Since, this attribute V_A is dropped after comparing the values, this consumption of additional space is temporal.

CONCLUSION

The method proposed does not depend on existence of primary key. It never altered any of the values present and hence preserved the precision of data. Since the overhead space to accommodate this virtual attribute is required until it has been verified, it is negligible.

This study can further be extended for distributed databases which can be across multiple physical locations with appropriate handling of replication and duplication. Distributed database has more technologies like autonomy, synchronous and asynchronous distributed databases.

REFERENCES

Agarwal, R. and J. Kiernan, 2002. Watermarking relational databases. Proceedings of 28th VLDB Conference, (VLDBC'2002), Hong Kong, pp: 155-166.

Li, Y.J., H.P. Guo and S. Jajodia, 2004. Tamper detection and localization for categorical data using fragile watermarks. Proceedings of ACM Workshop on Digital Rights Management (DRM), (WDRM'2004), ACM, New York, USA., pp: 73-82.

Shehab, M., E. Bertino and A. Ghafoor, 2008. Watermarking relational databases using optimization based techniques. Proceedings of the IEEE Transactions on Knowledge and Data Engineering, January 2008, IEEE Computer Society, USA., pp: 116-129.

Zhang, Y., X.M. Niu and D.N. Zhao, 2004a. A method of protecting relational databases copyright with cloud watermark. Int. J. Inform. Technol., 1: 206-210.

Zhang, Y., X.M. Niu, J.M. Wang and D.Y. Li, 2004b. Watermarking relational databases using image. Proceedings of IEEE Conference on Machine Learning and Cybernetics, Aug. 26-29, Shanghai, China, pp: 1739-1744.

Zhang, Y., X.M. Niu, D. Wu, L. Zhao, J.C. Liang and W.J. Xu, 2006. A novel method of watermarking relational databases using character string. Proceedings of the 24th IASTED International Conference on Artificial Intelligence and Applications, (ICAIA'2006), Innsbruck, Austria, pp: 120-124.