Scientific Research Publishing

Engineering, 2024, 16, 309-323
https://www.scirp.org/journal/eng
ISSN Online: 1947-394X
ISSN Print: 1947-3931

Reversible Semi-Fragile Watermarking Technique for Integrity Control of Relational Database

Ali Hamadou[1*], Abdoul Aziz Issaka Hassane[1], Lanciné Camara[2], Harouna Naroua[3]

[1]Department of Mathematics, Dan Dicko Dankoulodo University of Maradi, Maradi, Niger
[2]Social Science and Management University of Bamako, Bamako, Mali
[3]Department of Mathematics and Computer Science, Abdou Moumouni University, Niamey, Niger
Email: *ali.hamadou@uddm.edu.ne abdoulaziz.issaka@uddm.edu.ne, lcamaralancine@yhoo.fr, hnaroua@yahoo.com

How to cite this paper: Hamadou, A., Hassane, A.A.I., Camara, L. and Naroua, H. (2024) Reversible Semi-Fragile Watermarking Technique for Integrity Control of Relational Database. *Engineering*, **16**, 309-323.
https://doi.org/10.4236/eng.2024.169023

Received: August 20, 2024
Accepted: September 26, 2024
Published: September 29, 2024

Abstract

Reversible watermarking schemes for relational database are usually classified into two groups: robust schemes and fragile schemes. The main limitation of existing reversible fragile methods is that they cannot differentiate between legal and malicious modifications. In this paper, we introduce a novel lossless semi-fragile scheme based on prediction-error expansion for content protection of relational database. In the proposed method, all attributes in a database relation are first classified according to their sensitivity to legitimate updates. Then, the watermark is embedded by expanding the prediction error of the two least significant digits of securely selected attributes. At watermark extraction, the proposed method has the ability to fully restore the original data while detecting and localizing tampering. The applicability of our method is demonstrated theoretically and experimentally.

Keywords

Semi-Fragile Watermarking, Integrity Control, Reversibility, Prediction-Error Expansion

1. Introduction

With the increasing popularity of sharing data stores across the Internet, digital technology faces the challenge of preventing piracy of precious and sensitive data in different formats such as multimedia [1] [2], and database [3]-[7]. Content owners allow their data to be accessed and used remotely, thereby exposing them to a threat of data theft. To tackle this situation, digital watermarking has been proposed as a promising technology for ownership proofing and data integrity

control. Reversible schemes for relational data are usually classified in two categories: robust and fragile. Robust schemes are used for copyright protection [8]-[12], while fragile schemes are designed for tamper detection and content integrity verification [10]-[14]. Yet, one of the main limitations of most tamper detection schemes is their complete fragileness, *i.e.*, even a change of a single bit is detected as tampering. As a result, any legal update of the watermarked data would be regarded as malicious modification. A relatively few semi-fragile schemes were proposed for watermarking relational database [13]-[15]. The downside with these techniques is that they are irreversible, meaning that they introduce permanent distortions in the data. Although these errors are claimed to do not compromise the data usefulness, they are not tolerated in sensitive applications such as military and medical database systems.

In this paper, we propose a new reversible blind and semi-fragile watermarking technique for authentication of relational data based on prediction-error expansion (PEE). The purpose of semi-fragile watermarking is to verify that the content has not been altered maliciously, while allowing modifications by legitimate updates. The contributions of the proposed scheme can be summarized as follows:

- It allows the content owner to verify the integrity of database by detecting malicious modifications made to the database, while allowing some necessary updates;
- It allows original data recovery at watermark detection;
- It preserves data usefulness by encoding the watermark bits in the prediction error of securely selected numeric attributes;
- It is fully blind, *i.e.* the original database is not required for watermark detection.

The rest of this paper is organized as follows: In Section 2, previous work is discussed. An overview of PEE and the preprocessing phase are introduced in Section 3. In Section 4, the watermark embedding and detection procedures are explained in detail. The security analysis of the new scheme is described in Section 5. In Section 6, the proposed method is evaluated experimentally, and a comparison with some related works is provided. The conclusion is given in Section 7.

2. Related Work

Reversible watermarking for integrity control is currently a great concern in medical and military relational database applications, where they contribute to sensitive decision-making.

In [13], Shah *et al.* proposed a semi-fragile watermarking scheme for relational database integrity verification. Besides detection and localization of database tampering, the proposed scheme allows modifications to the data that need periodic updates. Watermark embedding is performed by adjusting the text case of selected data values, thus preserving semantic meaning of data. However, this method cannot be applied to numeric relational data. Moreover, changes made to text attributes values are not reversible.

In [14], a semi-fragile watermarking technique for tamper detection of relational

databases is proposed. The scheme associates a secret weight to database attributes according to their sensitivity to benign updates. The watermark is embedded in the least significant bits (LSB) of high-weighted attributes. Intermediate-weighted attributes can be legitimately modified up to a predefined level of distortion, while the low-weighted ones can be altered without any constraint. Consequently, the scheme can detect and localize malicious modifications when allowing authorized updates. However, the original data cannot be recovered.

Murugan *et al.* proposed a zero-watermarking scheme for integrity checking of relational database [15]. In their method, all attributes and tuples are sorted to generate a new tuple by processing each attribute value of the existing tuples. The watermark, constructed using the hash values of the newly created tuple, the database relation and the secret key, is not actually encoded into the data. Instead, it is registered with a Certification Authority. The scheme is claimed to be semi-fragile, but only innocent attacks (tuple/attribute sorting) are allowed. In addition, the proposed algorithm is not reversible and requires re-watermarking in case of legal data modification.

Contreras *et al.* [16] [17] proposed a lossless watermarking based on circular interpretation of bijective modulations for integrity verification of medical relational databases. The proposed technique modulates the relative angular position of the circular histogram center of mass of one numeric attribute in the relation. The foregoing scheme is a dual solution as it can be used for copyright protection, integrity control or traitor tracing. Nevertheless, it does not allow any legal data update.

In [18], a reversible fragile watermarking (RFW) technique for tamper detection and original data recovery in relational database was presented. In the proposed work, two optimization techniques, namely Orthogonal Learning Particle Swarm Optimization (OLPSO) and Fire Fly (FA) are used to find the optimal locations in the database for embedding the watermark. In case of tampering, the method has the ability to restore the database back to the original state. One of the limitations of this method is its complete fragileness.

A fragile, blind and reversible method for tamper detection in decision systems is presented in [19]. Based on rough set theory, the introduced scheme, first prepares a secure signature by encoding the information on reducts, rules and their support values. Then, it securely embeds the signature into the dataset. In case of alteration in reducts, the proposed technique can recover the original value. However, no legal update is allowed.

In [20], a RFW based on PEE is presented. The proposed scheme has the ability to detect and localize tampering at attribute level, while restoring the original data. However, it does not allow any legal modification.

3. Preliminary

3.1. Prediction-Error Expansion

The prediction-error expansion (PEE) was first proposed by Thodi and Rodriguez

[21]. The mechanism of PEE-based embedding and extraction can be briefly described as follows:

Let x be the intensity of a pixel in a gray-scale image, and $\hat{x}$ its predicted value. For embedding a watermark bit i, the prediction-error $e = x - \hat{x}$ is expanded by shifting left its binary format so as to create a vacant LSB into which i is inserted. The expanded prediction-error e_w and the watermark value x_w are given by

$$e_w = 2e + i \tag{1}$$

$$x_w = \hat{x} + e_w = x + e + i \tag{2}$$

For watermark decoding, first, the prediction-error $e_w = x_w - \hat{x}$ is computed from the watermark image. Then, the embedded bit i is extracted from the LSB position of e_w. Next, the original prediction-error is obtained by $e = \left\lfloor \frac{e_w}{2} \right\rfloor$. Finally, the original pixel value is restored as follows:

$$x = x_w - e - i \tag{3}$$

For example, assume $x = 223$, $a = 208$, $b = 213$ et $c = 213$. Then, the prediction error is given by: $e = 223 - 208 = 15$. Let $i = 1$ be the embedding watermark bit. The expanded prediction error and the watermarked pixel values can be computed as follows: $e_w = 2 * 15 + 1 = 31$ and $x_w = 208 + 31 = 239$. At detection phase, the concealed bit and the original prediction error are given by: $i = \mathrm{LSB}(e_w) = 1$ and $e = \left\lfloor \frac{31}{2} \right\rfloor = 15$. Lastly, the original pixel value can be restored as follows: $x = 239 - 15 - 1 = 223$.

In this work, we assume that the original pixel x is the two least significant digits (LSDs) of any numerical attributes to be selected for watermarking, and its predicted intensity $\hat{x}$ is the relevant tuple primary key hash value, which is known at both embedding and detection phases.

3.2. Preprocessing

The main goal of our solution is to construct a reversible semi-fragile watermark in such a way that, besides detecting malicious modifications, it could allow some legal updates. A fragile watermark is usually constructed from attribute using one-way hash functions such as MD5 or SHA. It is the property of one-way hash functions that any minor change to the input will randomize the output. Since, different attributes might have different degrees of sensitivity to legitimate modification, we can obtain a database semi-fragile watermarking if we can categorize attributes according to their importance in the database, and define some criteria for their inclusion in the hash calculation.

Let $R\left(P_k, A_0, A_1, \cdots, A_{\gamma-1}\right)$ be the database relation to watermark. $A_0, A_1, \cdots, A_{\gamma-1}$ are all numeric attributes, and P_k is the primary key. The design of our reversible semi-fragile scheme is as follows:

Step 1. Attribute classification: Based on the sensitivity to benign updates, we define three classes of attributes, namely Sensitive attributes ($\mathcal{S}$), Semi-Sensitive

attributes (*SS*) and Non-Sensitive attributes (*NS*), such that $S \cup SS \cup NS = R$ and $S \cap SS \cap NS = \varnothing$. Class *S* contains attributes that do not need to be updated once they are recorded. Examples of such attributes include primary key, social security numbers (SSN), age, blood group, account number, etc. *SS* attributes are those that can be legitimately modified within certain boundaries, *i.e.*, up to a predefined level of distortion. For example, we can allow to update all parts of a phone number except the country code. If 00227******** is a phone number, then 00227 represents the country code and * denotes the updatable part. *NS* is the category of attributes that can be freely modified.

Step 2. Legal update constraints for *SS* attributes: In order to allow legal modification of *SS* attributes, we need to define the degree of distortion that could be tolerated. For this purpose, we divide each *SS* attribute into two parts: a "rough portion" (*SSR*) and a "flat portion" (*SSF*). The rough portion is the non-updateable part of the attribute. Any modification made to it should be detected as tampering. The flat portion of the attribute can be legitimately altered. For numeric attributes, it could be defined in terms of the number of least significant digits (LSDs) or least significant bits (LSBs) available for modification. Since different attributes may tolerate different levels of distortion, we define ε_k LSDs as legal alteration bandwidth for each attribute A_k of *SS* class.

Step 3. Attribute Sorting: In a database relation, the order of attributes is normally not important. However, for ensuring the synchronization between the embedded and the extracted watermarks, this order should be the same at both embedding and detection phases. To satisfy this requirement, we define a recoverable secret "initial" order of attributes as follows. First, a name hash value is computed for each attribute using a secret key only known to the owner. Then, the attributes are virtually sorted in ascending order of their name hash values. Since attribute names, usually, do not change in a database relation, it is clear that the secret order can be efficiently recovered at watermark detection. As a result, attribute sorting operation will not be considered as an attack.

4. Proposed Scheme

In this section, we describe in detail our semi-fragile watermarking technique. In one hand, the scheme should be robust to legitimate modifications and innocent (*i.e.*, sorting) attacks. In the other hand, it should detect and localize malicious modifications. Moreover, since PEE is used our proposed scheme, the original relational data will be easily recovered at watermark detection.

Table 1 summarizes the notations used in the rest of this paper.

Table 1. Notations.

Symbol	Description
η	Number of tuples in the relation
γ	Number of attributes in the relation

Continued

$t_i.A_j$	j^{th} attribute of the i^{th} tuple
$t_i.P_k$	Primary key of i^{th} tuple
h_i	Hash value of i^{th} tuple's primary key
H_i	Hash value of i^{th} attribute
K_s	Secret key
g	Number of groups
v	Average number of tuples in a group
G_j	j^{th} group
e	Prediction error
e_w	Expanded prediction error
W	Embedded watermark
W^*	Extracted watermark
a	Number of sensitive attributes

4.1. Data Grouping

For tamper localization, our strategy consists in virtually splitting the database relation into g secret groups, in which a watermark is embedded independently. The grouping technique is presented in **Algorithm 1**. The relation R is securely partitioned based on the primary key hash values of tuples. Using the property that secure hash functions generate uniformly distributed message digests, our partitioning method will create groups of roughly equal size. Notice that the sorting operation is also virtual. It is necessary to enforce some correlation between tuples.

Algorithm 1: Data grouping

1. **for each** tuple $t_i \in R$ **do**
2. $h_i = \text{HMAC}(K_s//t_i.P_k//K_s)$ // Primary key hash
3. $j = h_i \bmod g$ // group index
4. insert tuple t_i into group G_j
5. **end for**

sort all tuples in G_j in increasing order of their primary key hash

6. return $(G_1, G_2, \ldots, G_g)$

4.2. Watermark Embedding

Usually, the more important an attribute is, the less it may be legally modified after watermark insertion. For this reason, in other to ensure reversibility, we choose to embed the semi-fragile watermark by expanding the prediction-error of

the two LSDs of sensitive attributes. Since the distortions caused by this process are minor and not permanent, the data usefulness will not be compromised. The watermark embedding process in a given group, depicted in **Algorithm 2**, is explained as follows. First, a watermark is generated from the hash values of tuples as shown in **Algorithm 3**. Note that, *NS* attributes and *SSF* data are excluded in the hash calculation. Afterwards, the prediction-error of the selected sensitive attribute is expanded with the corresponding watermark bit. Finally, the new value of the two LSDs is combined with the corresponding most significant digits (MSD) to update the value of the watermarked attribute.

Algorithm 2: Watermark embedding

1. W = GenWM ( ) //Watermark computation: see **Algorithm 3**
2. **for each** tuple t_i **do**
3. select S attribute with index $j = h_i \bmod \alpha$
4. lsd = Get2LSD($t_i.A_j$) //Extract the 2 LSDs
5. T = GetMSD($t_i.A_j$) // Most significant digits of $t_i.A_j$
6. e = lsd - h_i //Prediction error
7. $e_w = 2^*e + W[i]$ // expanded prediction error
8. newval = $e_w + h_i$ // New value of the 2 LSDs
9. $t_i.A_j$ = To_number(T, newval) //Update the watermarked attribute value
10. **end for**

Algorithm 3: GenWM( )

1. h = HMAC(K_s|| h_0||h_1||…|| h_{v-1} || K_s) //Hashing of primary key hash values
2. **for each** tuple t_i **do**
3. H_i = HMAC(K_s||S|| SSR||K_s) //tuple hash
4. **end for**
5. H = HMAC(h|| H_0||H_1||…|| H_{v-1} || K_s) //Group hash
6. W = v MSB (H) //Watermark: assume length(H) >= v
7. return W

4.3. Watermark Detection and Data Recovery

The watermarked data may be exposed to attacks intending to modify the database content. An attack will succeed, if the pirate could maliciously modify S or SSR data, without disturbing the watermark. For watermark synchronization, it is important to restore the original order of attributes. In addition, as in the embedding stage, the relation is virtually divided into secret groups (**Algorithm 1**). For tamper detection, each group is authenticated independently. **Algorithm 4** outlines the steps involved in the detection process. First, a watermark W^* is extracted from the suspicious group as shown by lines 1 through 6. Then, the original attribute value is restored. Lastly, the integrity of the group is verified by comparing the generated watermark W with the extracted one as described from line 11 through 14. The detection process is fully blind because it does not require the knowledge of the original database relation.

Algorithm 4: Watermark detection and recovery

1. **for each** tuple t_i **do**
2. select S attribute with index $j = h_i \bmod \alpha$
3. lsd = Get2LSD($t_i.A_j$)
4. T = GetMSD($t_i.A_j$) // Most significant digits of $t_i.A_j$
5. e_w = lsd - h_i // watermarked prediction error
6. $W^*[i]$ = LSB(e_w) // recovering embedded bit
   // Data Recovery
7. $e = \left\lfloor \frac{e_w}{2} \right\rfloor$ // original prediction error
8. origlsd = lsd – e - $W^*[i]$ // original value of the 2 LSDs
9. $t_i.A_j$ = To_number(T, origlsd) // attribute original value
10. **end for**
    //Integrity verification
11. W = GenWM ( ) //Original watermark: see **Algorithm 3**
12. **if** $W \neq W^*$ **then**
13. group not authentic
14. **end if**

5. Security Analysis

The objective of our scheme is to detect malicious modifications, while allowing some legal data updates. The goal of an attacker is to make his modifications undetectable so as to fail the watermark verification. For this reason, we focus on the analysis of the error rate in tamper detection, which is the probability of successful attack. In other words, it is the probability that the embedded W matches the extracted watermark W^* after tampering. In this analysis, we discuss common database attacks: single modification and massive modifications.

5.1. Single Modification Attacks

There are three single modifications: alter an attribute value, insert a tuple, delete a tuple. Assume that the attacker modifies a single value in the database. If the modification is made to a sensitive attribute or the rough part of a semi-sensitive attribute, the relevant group hash value will be randomized. Since the embedded watermark W is extracted from group hash, the watermark will also be randomized. After the modification, each bit of the embedded watermark W has equal probability to match the corresponding bit in the extracted watermark W^*. Therefore, the error rate for detecting this attack is given by

$$P_e = \frac{1}{2^v} \tag{4}$$

where v is the watermark size, which is determined by the number of tuples in the group.

Now, suppose that the pirate inserts a tuple into the protected relation. Due to our grouping technique, the newly inserted tuple will fall into a single group, thus increasing the associated watermark size v. Therefore, the failure rate in detecting

this tampering is formulated as

$$P_e = \frac{1}{2^{v+1}} \tag{5}$$

Similarly, the deletion of a single tuple will reduce the affected group size. Consequently, the relevant watermark will be randomized. Since there are $v-1$ bits remaining in the watermark, we obtain

$$P_e = \frac{1}{2^{v-1}} \tag{6}$$

Figure 1 shows the probability of error for the three single modification attacks using various watermark sizes. We can easily see that, in all cases, the failure rate is very low, and it decreases exponentially when the watermark length increases.

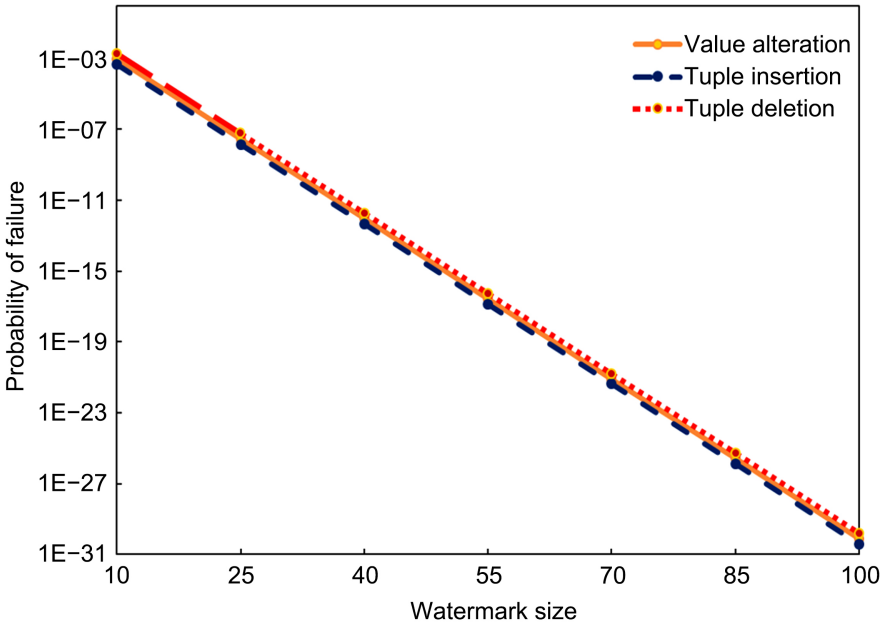

Figure 1. Failure rates for single modification attacks.

5.2. Massive Modification Attacks

This category of attacks includes multiple values alteration, multiple tuples insertion, and multiple tuples deletion.

Consider the scenario where multiple attribute values are maliciously modified. Assume that $k\left(1 \le k \le g\right)$ groups are affected. Since the probability that our scheme fails to detect a single value change in any group is $\frac{1}{2^v}$, the overall probability that the modification does not affect any group watermark is obtained by

$$P_e = \frac{1}{2^{vk}} \tag{7}$$

Now, let $n\left(n \ge 2\right)$, be the number of inserted records. After grouping, each new tuple has equal probability to be assigned to any group. Assume that

$k\left(k \le g\right)$ groups are affected. When $n > k$, an average of $\frac{n}{k}$ tuples are added to each group. Thus, the failure rate for any group is $\frac{1}{2^{v+\frac{n}{k}}}$. Therefore, the probability that all embedded watermarks remain unchanged after the attack is given by

$$P_e = \left(\frac{1}{2^{v+\frac{n}{k}}}\right)^k = \frac{1}{2^{vk+n}} \tag{8}$$

As for multiple tuples insertion, the deletion of $n\left(n \ge 2\right)$ tuples could affect several groups. In this case, if $k\left(k \le g\right)$ groups are changed, we have

$$P_e = \frac{1}{2^{vk-n}} \tag{9}$$

The failure probabilities for massive attacks are all monotonic decreasing with the watermark size v and the number of relevant watermarks k. This is illustrated in Figure 2 using different watermark sizes. For simplicity, we set $k = 5$ and $n = 20$. Here again, we can see the error rates are extremely low.

On the basis of the theoretical results, the proposed scheme is efficient enough to detect and localize traditional attacks.

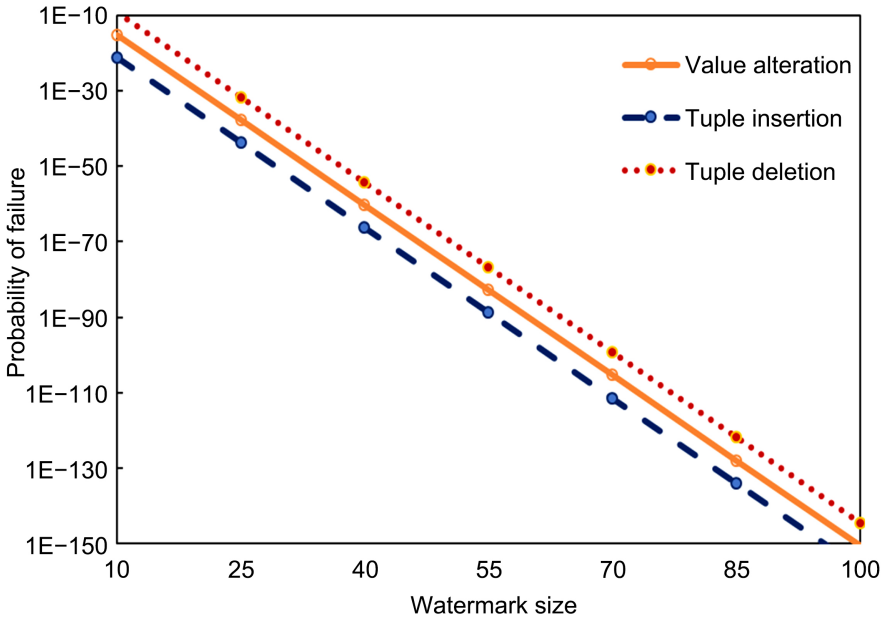

Figure 2. Failure rates for massive modification attacks.

6. Experimental Results

Our experiments were conducted on an Intel core i7-10510U CPU, 2.3 GHz with 16 GB RAM, running SQL Server 2016 and Microsoft Visual C#. For testing purpose, we used a generated database relation, which consists of 10 numerical attributes $id, A_1, \cdots, A_9$, (id is the primary key) and 10,000 tuples. The attributes

were categorized as follows: $S=\{id, A_1, A_3, A_6, A_9\}$, $SS=\{A_2, A_5, A_8\}$ and $NS=\{A_4, A_7\}$. For watermark computation, we used SHA-1 as one-way hash function [22].

We tested the efficiency of our scheme in tamper detection and integrity checking. Experiments were performed for the three massive attacks. For each kind of experiment, we varied the watermark size v (*i.e.* the group size) from 10 to 90 with an increment of 40. Each experiment was repeated 10 times, and the average results were recorded as the detection rate. It is important to note that the detection rate is complementary to the failure rate. The detection probability is the percentage of correctly detected modifications.

6.1. Multiple Values Alteration

Two different experiments were performed for this attack: 1) alteration of sensitive data, and 2) alteration of non-sensitive data. For each experiment, we randomly modified 10% to 90% of the values with varying watermark sizes.

In the first case, we only modified values from sensitive attributes (S) and rough portion of semi-sensitive attributes (SSR). The results are displayed in **Figure 3**. We observe that:

- The detection rate increases with an increase in the alteration rate and the group size (v);
- Very few alterations are missed when the modification rate and the group size are both small;
- For $v \geq 50$, 100% detection can be achieved.

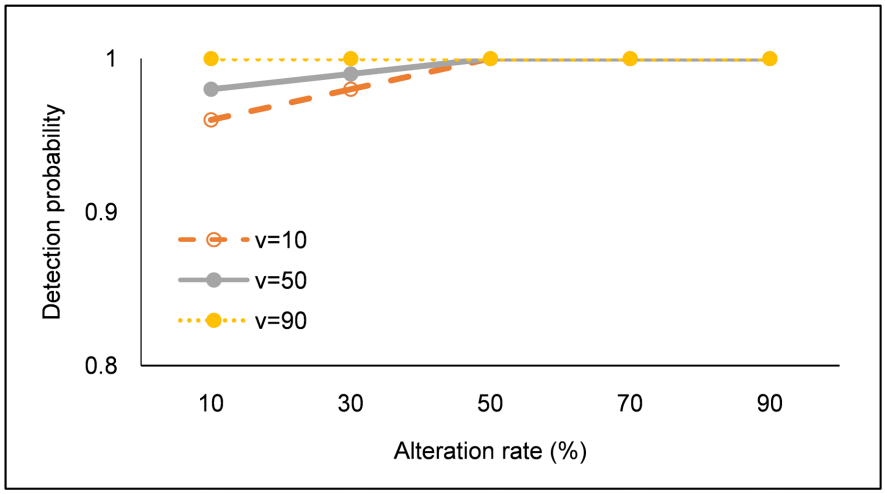

Figure 3. Detection rate (%) for sensitive data alteration.

In the second case, only non-sensitive values (NS) and flat part of semi-sensitive attributes (SSF) were altered. The results are shown in **Figure 4**. We observe that, no matter what are the alteration rate and the watermark size v, the detection rate is 0%. This is obvious, because the performed modifications are legitimate. Therefore, this result confirms the semi-fragileness of our scheme.

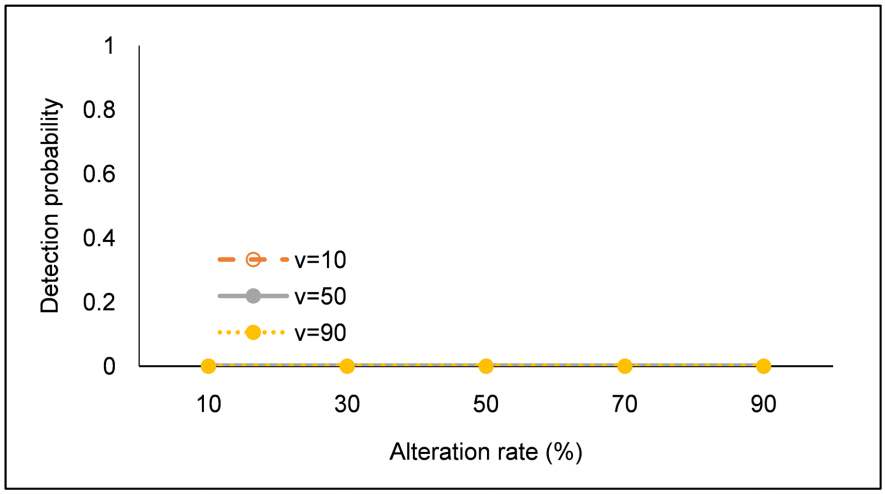

Figure 4. Detection rate (%) for *NS* and *SSF* data alteration.

6.2. Multiple Tuples Deletion

Figure 5 shows the results for multiple tuples deletion. We performed the experiments with varying deletion rates and watermark sizes. For small groups, we can observe that:

- The detection rate decreases with an increase in the deletion rate while it increases with an increase in the group size. This result is normal because with 90% deleted tuples, it is highly possible that the majority of the groups were completely deleted;
- 100% detection rate can be achieved even when the deletion rate is high, for large groups (e.g. $v = 90$).

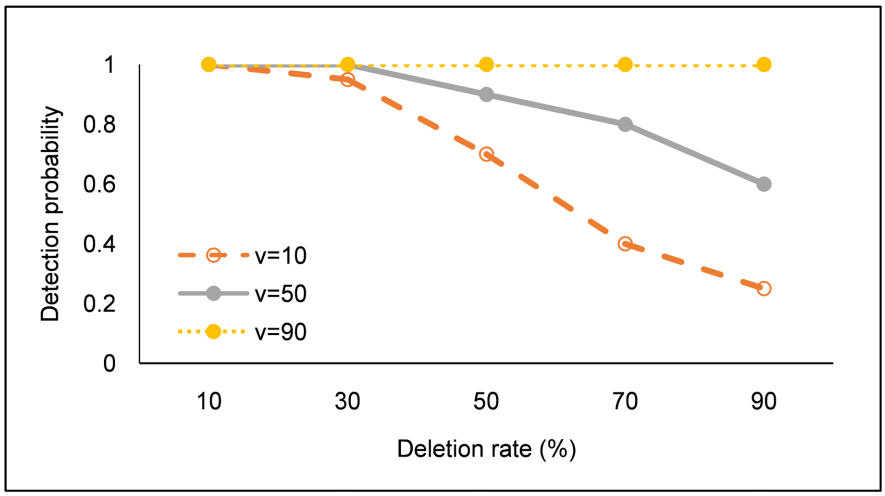

Figure 5. Detection rate (%) for multiple tuples deletion.

6.3. Multiple Tuples Insertion

To simulate this attack, we randomly and repeatedly inserted fake tuples into the watermarked groups. As a result, because of the fragileness of our scheme

regarding illegal change, all relevant hash values and watermarks are randomized. As shown in Figure 6, we observe that the proposed method can fully detect tampered groups for different values of *v*.

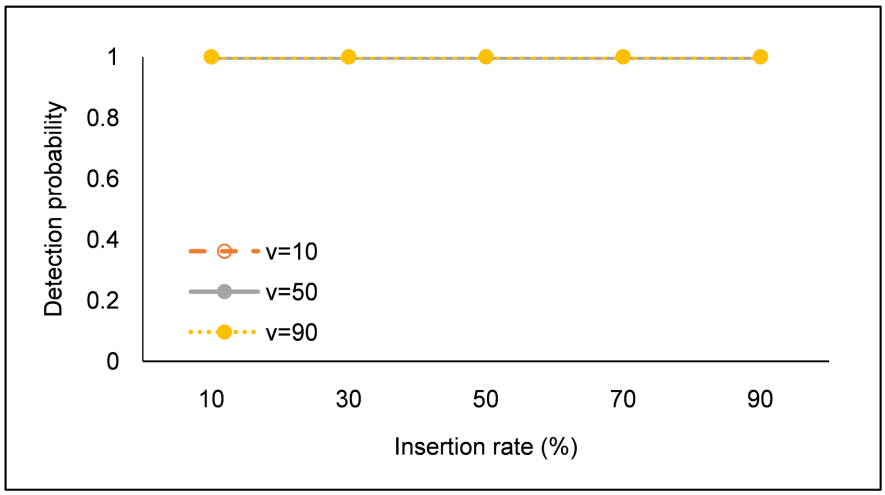

Figure 6. Detection rate (%) for multiple tuples deletion.

6.4. Comparison with Related Works

In Table 2, we present a comparison of our scheme with three most relevant recent works. We consider the following features: embedding channel (*i.e.*, data type), data updateability (*i.e.* legitimate modification), innocent attacks (*i.e.* tuple/attribute sorting), tamper detection, and reversibility (*i.e.* data recovery). We can easily see that, at the difference of other techniques, our scheme can allow data updateability, while enabling data restoration and tamper detection.

Table 2. Comparison with previous works.

Feature	Shah *et al.* [13]	Murugan *et al.* [15]	Khanduja *et al.* [19]	Proposed scheme
Data type	Non-numeric	Any	Numeric	Numeric
Legitimate modification	Yes	No	No	Yes
Innocent attacks	Not resilient	Resilient	Not resilient	Resilient
Reversibility	No	No	Yes	Yes
Tamper detection	Yes	Yes	Yes	Yes

7. Conclusion

In this paper, a novel reversible semi-fragile watermarking method for integrity checking of relational databases is presented. To ensure data updateability, the database attributes are classified as sensitive, semi-sensitive and non-sensitive. The semi-fragile watermark is generated from the data characteristics to detect

malicious modifications, while allowing legal updates. Innocent attacks such as tuple/attribute sorting are not considered as tampering. Moreover, due to the use of PEE, the scheme has the ability to recover back the original data from any authentic group. From the theoretical analysis and experimental results obtained, it was proved that the probability of detecting and localizing common attacks is extremely high.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Aberna, P. and Agilandeeswari, L. (2023) Digital Image and Video Watermarking: Methodologies, Attacks, Applications, and Future Directions. *Multimedia Tools and Applications*, **83**, 5531-5591. https://doi.org/10.1007/s11042-023-15806-y

[2] Gao, Z.Z., Cheng, Y. and Yin, Z.X. (2024) Fragile Model Watermarking: A Comprehensive Survey of Evolution, Characteristics, and Classification.

[3] Rani, S. and Halder, R. (2022) Comparative Analysis of Relational Database Watermarking Techniques: An Empirical Study. *IEEE Access*, **10**, 27970-27989. https://doi.org/10.1109/access.2022.3157866

[4] Hamadou, A., Sun, X., Gao, L. and Shah, S.A. (2011) A Fragile Zero-Watermarking Technique for Authentication of Relational Databases. *International Journal of Digital Content Technology and its Applications*, **5**, 189-200. https://doi.org/10.4156/jdcta.vol5.issue5.21

[5] Camara, L., Li, J., Li, R. and Xie, W. (2014) Distortion-Free Watermarking Approach for Relational Database Integrity Checking. *Mathematical Problems in Engineering*, **2014**, 1-10. https://doi.org/10.1155/2014/697165

[6] Camara, L., Coulibaly, D., Hamadou, A. and Li, J. (2017) An Effective Approach for Non-Numeric Relational Database Verification. *International Journal of Database Theory and Application*, **10**, 35-46. https://doi.org/10.14257/ijdta.2017.10.6.03

[7] Gao, L., Wang, D. and Hamadou, A. (2013) New Fragile Database Watermarking Scheme with Restoration Using Reed-Solomon Codes. *Journal of Computational and Theoretical Nanoscience*, **10**, 147-153. https://doi.org/10.1166/jctn.2013.2671

[8] Li, D., Ma, C., Gao, H. and Jin, X. (2023) LBP Feature and Hash Function Based Dual Watermarking Algorithm for Database. *Data & Knowledge Engineering*, **148**, Article 102228. https://doi.org/10.1016/j.datak.2023.102228

[9] Lin, C., Nguyen, T. and Chang, C. (2021) LRW-CRDB: Lossless Robust Watermarking Scheme for Categorical Relational Databases. *Symmetry*, **13**, Article 2191. https://doi.org/10.3390/sym13112191

[10] Hou, R., Xian, H., Wang, X. and Li, J. (2019) A Robust Reversible Watermarking Scheme for Relational Data. In: Li, J., Liu, Z.L. and Peng, H., Eds., *Security and Privacy in New Computing Environments*, Springer, 545-550. https://doi.org/10.1007/978-3-030-21373-2_44

[11] Hu, D., Zhao, D. and Zheng, S. (2019) A New Robust Approach for Reversible Database Watermarking with Distortion Control. *IEEE Transactions on Knowledge and Data Engineering*, **31**, 1024-1037. https://doi.org/10.1109/tkde.2018.2851517

[12] Chang, C., Nguyen, T. and Lin, C. (2013) A Blind Reversible Robust Watermarking

Scheme for Relational Databases. *The Scientific World Journal*, **2013**, Article 717165. https://doi.org/10.1155/2013/717165

[13] Arif Shah, S., Ali Khan, I., Hassan Kazmi, S.Z. and Binti Md Nasaruddin, F.H. (2021) Semi-Fragile Watermarking Scheme for Relational Database Tamper Detection. *Malaysian Journal of Computer Science*, **34**, 1-12. https://doi.org/10.22452/mjcs.vol34no1.1

[14] Hamadou, A., Sun, X., Shah, S. and Gao, L. (2011) A Weight-Based Semi-Fragile Watermarking Scheme for Integrity Verification of Relational Data. *International Journal of Digital Content Technology and its Applications*, **5**, 148-157. https://doi.org/10.4156/jdcta.vol5.issue8.17

[15] Murugan, R., John, A.T. and Ibrahim, S. (2020) A Semi-Fragile Watermarking Scheme for Integrity Checking of Relational Databases. *International Journal of Recent Technology and Engineering*, **8**, 806-812. https://doi.org/10.35940/ijrte.f6996.038620

[16] Franco-Contreras, J., Coatrieux, G., Cuppens, F., Cuppens-Boulahia, N. and Roux, C. (2014) Robust Lossless Watermarking of Relational Databases Based on Circular Histogram Modulation. *IEEE Transactions on Information Forensics and Security*, **9**, 397-410. https://doi.org/10.1109/tifs.2013.2294240

[17] Franco Contreras, J., Coatrieux, G., Chazard, E., Cuppens, F., Cuppens-Boulahia, N. and Roux, C. (2012) Robust Lossless Watermarking Based on Circular Interpretation of Bijective Transformations for the Protection of Medical Databases. 2012 *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, San Diego, 28 August-1 September 2012, 5875-5878. https://doi.org/10.1109/embc.2012.6347330

[18] Unnikrishnan, K. and Pramod, K.V. (2017) Robust Optimal Position Detection Scheme for Relational Database Watermarking through HOLPSOFA Algorithm. *Journal of Information Security and Applications*, **35**, 1-12. https://doi.org/10.1016/j.jisa.2017.04.005

[19] Khanduja, V. and Chakraverty, S. (2018) Fragile Watermarking of Decision System Using Rough Set Theory. A*rabian Journal for Science and Engineering*, **43**, 7621-7633. https://doi.org/10.1007/s13369-018-3120-7

[20] Hamadou, A., Camara, L., Issaka Hassane, A.A. and Naroua, H. (2020) Reversible Fragile Watermarking Scheme for Relational Database Based on Prediction-Error Expansion. *Mathematical Problems in Engineering*, **2020**, 1-9. https://doi.org/10.1155/2020/1740205

[21] Thodi, D.M. and Rodriguez, J.J. (2004) Prediction-Error Based Reversible Watermarking. 2004 *International Conference on Image Processing*, Singapore, 24-27 October 2004, 1549-1552. https://doi.org/10.1109/icip.2004.1421361

[22] Ferguson, N. and Schneier, B. (2003) Practical Cryptography. Wiley.