Reversible And Blind Database Watermarking Using Difference Expansion

Gaurav Gupta Josef Pieprzyk

Centre for Advanced Computing - Algorithms and Cryptography,
Department of Computing,
Macquarie University,
Sydney, NSW 2109, Australia
{ggupta,josef}@ics.mq.edu.au

ABSTRACT

Database watermarking has received significant research attention in the current decade. Although, almost all watermarking models have been either irreversible (the original relation cannot be restored from the watermarked relation) and/or non-blind (requiring original relation to detect the watermark in watermarked relation). This model has several disadvantages over reversible and blind watermarking (requiring only watermarked relation and secret key from which the watermark is detected and original relation is restored) including inability to identify rightful owner in case of successful secondary watermarking, inability to revert the relation to original data set (required in high precision industries) and requirement to store unmarked relation at a secure secondary storage. To overcome these problems, we propose a watermarking scheme that is reversible as well as blind. We utilize difference expansion on integers to achieve reversibility. The major advantages provided by our scheme are reversibility to high quality original data set, rightful owner identification, resistance against secondary watermarking attacks, and no need to store original database at a secure secondary storage.

Keywords

database, watermarking, reversible, copyright

1. INTRODUCTION

Electronic communication, faster internet data transfer speed, and peer-to-peer communication facilitate convenient transfer of multimedia objects. However, they also open up the possibility of copyright violations. Publishers need to insert ownership mark in the media object to discourage users from illegally downloading multimedia and thereby protect copyright. This process is referred to as *watermarking*. The major requirements of a watermarking algorithm are that the watermark should not be noticeable (imperceptibility), watermark should survive possible attacks (robustness), successful recognition and extraction of watermark in a watermarked copy, watermark detection should require only the watermarked copy and a secret key (blindness), and a sufficiently large watermark should be insertable in the multimedia object (high capacity).

E-FORENSICS 2008, January 21-23, Adelaide, Australia

DOI 10.4108/e-forensics.2008.2691

Images, video, audio, software, natural language documents, and databases are the usual candidates for watermarking. Images are the primary contenders [4, 5, 8, 9, 10] given that changing the characteristics of pixel does not substantially degrade image quality and the watermarking capacity in millions of pixels is very high. The insertion algorithm selects the pixels that will carry watermark using pseudo random generators with a secret seed. Thus an attacker's task is made even harder by requiring him to find out the watermark location.

Comparatively, database watermarking is a new field where research interest has risen recently [1, 2, 16, 11, 12, 14, 15, 18, 19, 20]. A typical database watermarking scenario is when a publisher $\mathcal{C}$ creates a database relation $\mathcal{R}$ and sells it to $\mathcal{O}$. If $\mathcal{O}$ is a traitor, it illegally sells the relation to others. To prevent this, $\mathcal{C}$ embeds a watermark $\mathcal{W}$ in $\mathcal{R}$. Similarly, if a data provider $\mathcal{D}$ uploads relation $\mathcal{R}$ for remote query process, an attacker might reconstruct the original relation by assembling query results. Hence, $\mathcal{D}$ uploads a watermarked relation.

A blind watermarking scheme requires only watermarked object and a secret key to detect watermark while a non-blind watermarking scheme requires the unmarked multimedia object in addition to the first two inputs. The major disadvantage of a non-blind watermarking scheme is that one needs to store the unmarked object at a secure secondary storage location and feed it back to the detection algorithm later.

Reversible watermarking provides a mechanism to revert the watermarked relation back to the original unmarked relation using a secret key. The key advantages of reversibility are,

1. Allows for trial version of multimedia content, that can be later upgraded to the full version by reversing it. As an example, a company may want to distribute low quality (in terms of usability and precision) relations free of cost and then require customers to purchase a key using which they can revert the relation to high quality original relation. This is not facilitated by irreversible watermarking schemes.

2. Allows to introduce higher distortion in the data since original data can be regenerated by reversing the watermarking.

In this paper, we determine the requirements of database watermarking model, feasible attacks, and propose a reversible and blind database watermarking scheme addressing these concerns.

1.1 Organization of paper

We organize our papers as follows: Section 2 contains related work. Section 3 gives model of the adversary and describes potential attacks against database watermarking schemes. We present our watermarking scheme in Section 4, discuss our experimental results in Section 5 and analyze the model in Section 6 in terms of capacity and security in Section 6.1 and 6.2 respectively. The paper is concluded in Section 7 with a note on future direction.

1.2 Notations

The following notations are used in the paper,

- R : relation,
- r : tuple,
- $r.A_i$: i^{th} attribute in tuple r,
- $r.A_i^j$: j^{th} LSB of i^{th} attribute in tuple r,
- $r.P$: primary key of tuple r,
- $\circ$: concatenation,
- $\mathcal{H}()$: one-way hash function,
- $R \xrightarrow{ins(p)} R_w$: R_w is the watermarked relation upon party p watermarking relation R,
- $R_w \xrightarrow{det(p)} R$: the original relation R is restored by the party p from the watermarked relation R_w,
- $\lfloor x \rfloor$: the greatest integer smaller than x (floor function),
- $\lceil x \rceil$: the smallest integer greater than x (ceiling function),
- $size(x)$: size of x in bits,
- $abs(x)$: absolute value of x.

2. RELATED WORK

Majority of the database watermarking schemes rely on unique identification of tuple from the primary key value and the assumption that attacker cannot change the primary key without compromising usability. One of the first database watermarking algorithms was provided in [1]. The algorithms select one out of γ tuples for watermarking from a relation containing a total of η tuples. For each selected tuple r, an attribute A_i and a bit position j is secretly selected using a hash function computed on the combination of a private key and the tuple's primary key $\mathcal{F}(r.P) = \mathcal{H}(\mathcal{K} \circ \mathcal{H}(\mathcal{K} \circ (r.P))$, where $\mathcal{H}$ is the hash function, $\mathcal{K}$ is the secret key and $r.P$ is the primary key of the tuple r. The bit $r.A_i^j$ is then replaced by the LSB of $\mathcal{H}(\mathcal{K} \circ r.P)$. Inputs to the insertion algorithm are relation R containing υ attributes and η tuples, the fraction of tuples to be watermarked γ, the number of LSBs to be considered for watermarking ξ, and secret key $\mathcal{K}$. The insertion and detection algorithms from [1] are given in Algorithm 1 and Algorithm 2 respectively.

Input: Relation R, private key K, fraction $\frac{1}{\gamma}$, LSB usage ξ
Output: Watermarked relation R_w

forall *tuple* $r \in R$ **do**
  if $\mathcal{F}(r.P)\%\gamma = 0$ **then**
    $i = \mathcal{F}(r.P)\%\upsilon$;
    $j = \mathcal{F}(r.P)\%\xi$;
    $r.A_i^j = \mathcal{H}(\mathcal{K} \circ r.P)\%2$;
  end
end
return R;

Algorithm 1: Agrawal-Kiernan watermark insertion algorithm

Input: Watermarked Relation $\tilde{R}_w$, private key K, fraction $\frac{1}{\gamma}$, LSB usage ξ
Output: Detection Status $\in \{true, false\}$

$totalcount = matchcount = 0$;
forall *tuple* $\tilde{r}_w \in \tilde{R}_w$ **do**
  if $\mathcal{F}(r.P)\%\gamma = 0$ **then**
    $i = \mathcal{F}(r.P)\%\upsilon$;
    $j = \mathcal{F}(r.P)\%\xi$;
    if $\tilde{r}_w.A_i^j = \mathcal{H}(\mathcal{K} \circ \tilde{r}_w.P)\%2$ **then**
      $matchcount = matchcount + 1$;
    end
    $totalcount = totalcount + 1$;
  end
end
$\tau = min\{\theta : \mathcal{B}(\theta, totalcount, 1/2) < \alpha\}$; // $\mathcal{B}$ defined in Equation 1
if $matchcount \geq \tau$ **then**
  return $true$;
end
return $false$;

Algorithm 2: Agrawal-Kiernan watermark detection algorithm

The probability of having at least k successes from n trials is given in Equation 1, where probability of success in a single trial is p.

$$\mathcal{B}(k,n,p) = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} \quad (1)$$

Detection of τ or more bits results in a successful recovery of watermark. Hence, probability of τ out of ω ($\omega = \eta/\gamma$) bits being detected in a random database relation by chance ($B(\tau, \omega, \frac{1}{2})$) should be less than α. Thus α is the upper bound of the false positive probability.

Other papers in the field also make the same assumption (that the primary key cannot be changed by the attacker) as otherwise it would be (probably) impossible to uniquely identify the tuples carrying watermark [2, 16, 11, 12, 14, 15, 18, 19, 20]. The major drawbacks of this watermarking algorithm proposed by Agrawal and Kiernan are irreversibility, that renders object susceptible to secondary watermarking, and inconsideration for attributes' individual bandwith. Different attributes have varying watermarking carrying capacity. Thus individual modification limit ξ should be assigned to each attribute.

Several reversible and blind image watermarking schemes have been proposed. *Data compression* based reversal [6] compresses the least significant bits (LSBs) of n pixels selected into m bits where $m < n$. These m bits and $n - m$ watermark bits are then inserted in the n selected pixels. However, data compression based

watermarking schemes are extremely fragile since the lossless algorithms are not modification-resistant. *Histogram shifting* techniques [7] exploit the notion that neighboring pixels have high correlation and depending on the watermark bit, the histogram bins are circularly upgraded (if watermark=1) or downgraded (if watermark=0). Since database relation values do not possess correlation similar to images, histogram shift technique is irrelevant for our purpose. *Difference expansion* based watermarking [3, 17] integrates a watermark bit to an n-element vector such that the original vector and the watermark bit can be retrieved from the modified vector.

Difference expansion based watermarking performs invertible arithmetic operations on integers. A scheme to embed $n-1$ watermark bits in n vectors is given in [3] and a specific case for $n=2$ is described in [17] called pairwise difference expansion. For simplicity, we will introduce the latter scheme in this paper.

Given two adjacent pixels' values from a grayscale image, we compute average a and difference d as

$$a = \lfloor \frac{(x+y)}{2} \rfloor, d = x - y \tag{2}$$

This operation is invertible as x and y can be computed from a and d as follows:

$$x = a + \lfloor \frac{(d+1)}{2} \rfloor, y = a - \lfloor \frac{d}{2} \rfloor \tag{3}$$

The integer d is now changed to $d' = 2*d+b$ and x', y' are computed from a, d' and watermark bit b to be inserted as,

$$x' = a + \lfloor \frac{(d'+1))}{2} \rfloor, y' = a - \lfloor \frac{d'}{2} \rfloor \tag{4}$$

The new pixel values are x', y'. One can re-calculate a, d' from x', y' using Equation 2. The watermark bit is simply the LSB of d' and $d = \lfloor \frac{d'}{2} \rfloor$. Now from a, d one can compute the values of x, y using Equation 3. As a working example, consider two pixels $x = 106, y = 100$. From Equation (1), $a = 103, d = 6$, assuming $b = 1$, $d' = 2*6+1 = 13$. $x' = 103 + \lfloor (13+1)/2 \rfloor = 110, y' = 103 - \lfloor (13/2) \rfloor = 97$. Hence, the new pixel values are $x' = 110, y' = 97$. At the receiver's end $a = \lfloor (110+97)/2 \rfloor = 103, d' = 110 - 97 = 13$. $b = lsb(d') = 1, d = \lfloor d'/2 \rfloor = 6$. $x = 103 + \lfloor (6+1)/2 \rfloor = 106, y = 103 - \lfloor (6/2) \rfloor = 100$. Thus we can successfully recover the watermark bit and original pixel values from the modified values.

We denote the process of reversing a pair as $\{x_r, y_r\} = Reverse\{x, y\}$.

3. MODEL OF ADVERSARY

The set of possible attacks a watermark should survive are as follows.

A1: Random bitwise flipping attacks, i.e. some bits selected at random (probably with uniform probability distribution) are modified.

A2: Subtractive attack, i.e. some tuples chosen at random are deleted.

A3: Sorting, i.e. some tuples and/or attributes are chosen at random and their positions are changed. An ordering criteria maybe chosen by the attacker and the relation is then sorted in ascending/ descending order based on that criteria thereby resulting in a differently sorted relation.

A4: Secondary watermarking, i.e. a watermark is superimposed on the watermarked relation.

The degree of secrecy and randomness in selecting the tuples and attributes that will be marked along with proportion of the tuples selected for marking determines the security level of watermark against the attacks **A1** and **A2**. The assumption that primary key cannot be modified by the attacker ensures that attack **A3** is not successful since the correct order can be re-established using primary key values (for example, sorting tuples in ascending order of primary key). We focus on providing security against secondary watermarking.

Assume that Alice watermarks a relation R to create watermarked relation R_a. An attacker Mallory might make some modifications in R_a before re-watermarking it with a secondary watermark to create relation R_m, Watermarks of Mallory and Alice are detected in R_m with probabilities 1 and p respectively. Thus with probability $1-p$, the incorrect owner will be output and there is confusion over ownership with probability p. The problem can be averted by designing reversible watermarking algorithms as explained below.

Considering the same situation again when Alice and Mallory both watermark a relation, when the judge needs to determine the rightful owner, he asks both Alice and Mallory to detect their watermarks in their watermarked documents R_a and R_m, respectively. They reverse their relations to the original documents R and R'_a, respectively (as Mallory might have made some modifications in R_a before inserting the watermark). Alice's watermark is detected in the reversed relation of Mallory but Mallory's watermark is not detected in the reverse relation of Alice which proves that the sequence of watermarking was Alice followed by Mallory and thus establishes Alice as rightful owner. Recently a reversible scheme for database watermarking was proposed in [13]. The inserting algorithm stores the original bits that are later modified in an *embed map*. During the detection algorithm, the marked bits are sequentially replaced by bits from *embed map*. This approach suffers from the following drawbacks,

- if the adversary deletes one of the tuples, then the bits from the tuples positioned after the deleted ones will be distorted,
- the scheme is essentially non-blind since the information about the watermark needs to be stored in a safe location, and,
- incremental watermarking: If the database has to be updated and re-watermarked, one needs to reverse the entire relation.

Thus the main objective of our scheme is to eliminate these three shortcomings of [13] and still provide security against secondary watermarking attacks.

4. PROPOSED SCHEME

We intend to satisfy the major requirements of a watermarking scheme (as mentioned in Section 1) and at the same time facilitate reversibility of the watermarked relation. Difference expansion is the most suitable method to facilitate reversibility in database watermarking since the markable data is in numeric format. In order to utilize the reversible watermarking based on difference expansion, we select two attributes from the same tuple to carry the watermark

bit (call the two attributes A_i and A_j). We need to select the two attributes so that the distortion (change in attributes' values) is within the bounds. We ensure that the distortion is tolerable by placing an upper bound of ξ_i on the number of modifiable LSBs for attribute A_i. Let the tuple selected for watermarking be r and the attributes be A_i, A_j. The bit embedded is $lsb(\mathcal{H}(\mathcal{K} \circ r.P))$. Thus, when the detection algorithm is run and bit is extracted, it is compared to $lsb(\mathcal{H}(\mathcal{K} \circ r.P))$ for determining successful recovery. Since the attacker cannot modify the primary key, $lsb(\mathcal{F}(r.P))$ enables us to identify marked tuples and difference expansion facilitates reversal. The insertion and detection algorithms are provided in Algorithm 3 and Algorithm 4 respectively.

In lines 6, 7 of Algorithm 3, we ensures that in case the unmarked attributes were reversed, the difference between the reversed values and unmarked values should exceed distortion tolerance. This condition can detect the attributes which are not marked because of exceeding distortion limits. The condition is rechecked in lines 12, 13 of Algorithm 4 once the unmarked values are computed from the marked attributes.

In the detection algorithm, we also check that a significant proportion of marks are detected in the multimedia object in order to establish beyond reasonable doubt that the object is in fact watermarked. The significance level can be determined by parameter α as in [1]. We use percentage of marks detected, *prctng*, as a simpler and equally strict significance level metric. Considering *prctng* to ensure mark presence reduces the chances of false positives if *prctng* is sufficiently large (experimental results show 85% and over is desirable).

Input: Relation R, private key K, fraction γ, number of markable attributes υ, LSB usage $\Xi = \{\xi_1, \xi_2, \ldots, \xi_\upsilon\}$
Output: Watermarked relation R_w

1 **forall** *tuples* $r \in R$ **do**
2 **if** $\mathcal{F}(r.P)\%\gamma = 0$ **then**
3 $i = \mathcal{F}(r.P)\%\upsilon$; // identify attribute 1
4 $j = \mathcal{F}(\frac{r.P}{2})\%\upsilon$; // identify attribute 2
5 $x = max(A_i, A_j)$, $y = min(A_i, A_j)$;
6 $\{x_r, y_r\} = Reverse\{x, y\}$;
7 **if** $abs(x - x_r) > \xi_1$ *OR* $abs(y - y_r) > \xi_2$ **then**
8 $a = \lfloor \frac{x+y}{2} \rfloor$, $d = x - y$;
9 $b = lsb(\mathcal{H}(\mathcal{K} \circ r.P))$; // bit to embed
10 $d' = 2 * d + b$;
11 $x' = a + \lfloor \frac{d'+1}{2} \rfloor$, $y' = a - \lfloor \frac{d'}{2} \rfloor$;
12 **if** $A_i > A_j$ **then**
13 $\delta_1 = abs(A_i - x')$;
14 $\delta_2 = abs(A_i - y')$;
15 **if** $\delta_1 < \xi_i$ *AND* $\delta_2 < \xi_j$ **then**
16 $A_i = x', A_j = y'$;
17 **end**
18 **else**
19 $\delta_2 = abs(A_i - x')$, $\delta_1 = abs(A_i - y')$;
20 **if** $\delta_2 < \xi_i$ *AND* $\delta_1 < \xi_j$ **then**
21 $A_i = y', A_j = x'$;
22 **end**
23 **end**
24 **end**
25 **end**
26 **end**

Algorithm 3: Watermark insertion

Input: Watermarked Relation $\tilde{R}_w$, Secret parameter list $\phi = (\mathcal{K}, \gamma, \upsilon, \Xi), prcntg$
Output: {Watermark Status $\in \{true, false\}$, Restored Relation R}

1 $R = \tilde{R}_w$;
2 $matchcount = 0$; // matching watermark bits counter
3 $totalcount = 0$; // total watermark bits counter
4 **forall** *tuples* $\tilde{r}_w \in \tilde{R}_w$ **do**
5 **if** $\mathcal{F}(r_w.P)\%\gamma = 0$ **then**
6 $i = \mathcal{F}(r_w.P)\%\upsilon$; // identify marked attribute
7 $j = \mathcal{F}(\frac{r_w.P}{2})\%\upsilon$; // identify marked bit
8 $x' = max(A_i, A_j)$, $y' = min(A_i, A_j)$;
9 $a' = \lfloor \frac{x'+y'}{2} \rfloor$, $d' = x' - y'$;
10 $b = lsb(d')$ $d = \lfloor \frac{d'}{2} \rfloor$;
11 $x = a' + \lfloor \frac{d+1}{2} \rfloor$, $y = a' - \lfloor \frac{d}{2} \rfloor$;
12 $\{x_r, y_r\} = Reverse\{x, y\}$;
13 **if** $abs(x - x_r) > \xi_1$ *OR* $abs(y - y_r) > \xi_2$ **then**
14 **if** $A_i > A_j$ **then**
15 $\delta_1 = abs(A_i - x)$, $\delta_2 = abs(A_i - y)$;
16 **if** $\delta_1 < \xi_i$ *AND* $\delta_2 < \xi_j$ **then**
17 **if** $b = lsb(\mathcal{H}(\mathcal{K} \circ r.P))$ **then**
18 $A_i = x, A_j = y$;
19 $matchcount = matchcount + 1$;
20 **end**
21 $totalcount = totalcount + 1$;
22 **end**
23 **else**
24 $\delta_2 = abs(A_i - x)$, $\delta_1 = abs(A_i - y)$;
25 **if** $\delta_2 < \xi_i$ *AND* $\delta_1 < \xi_j$ **then**
26 **if** $b = lsb(\mathcal{H}(\mathcal{K} \circ r.P))$ **then**
27 $A_i = y, A_j = x$;
28 $matchcount = matchcount + 1$;
29 **end**
30 $totalcount = totalcount + 1$;
31 **end**
32 **end**
33 **end**
34 **end**
35 **end**
36 **if** $\frac{matchcount}{totalcount} \geq prctng$ **then**
37 return $\{true, R\}$;
38 **else**
39 return $\{false, \tilde{R}_w\}$;
40 **end**

Algorithm 4: Watermark detection

5. EXPERIMENTAL RESULTS

We carried out experiments with 1000 files having 200 to 300 tuples and 10 to 20 attributes each. The software generated the database files, inserted the watermark, made modifications of the watermarked relations, and detected the watermark in the attacked files. Changing fractions did not have major effect on detectability of watermark (with the exception when fraction=33%). As tolerance increases, probability of false positives increases and probability of detection also increases. With increasing attack levels, detection probability reduces and is confirmed by the experimental results. The worst case scenario occured when the attacker modified 48 out of every 100 tuples. In such a situation, 89 out of 100 times, the watermark was still detected corroborating the theoreti-

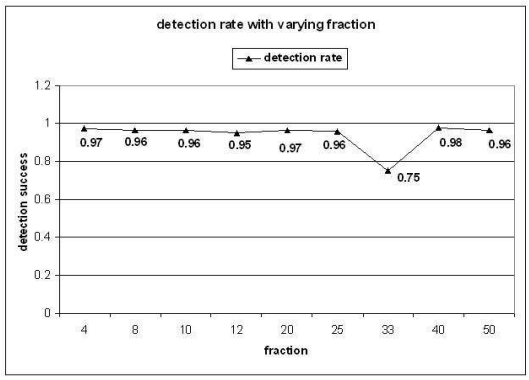

Figure 1: Effect of changing fraction of tuples marked on detection

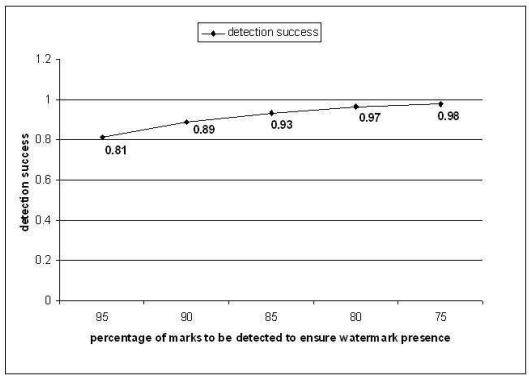

Figure 2: Effect of changing percentage of marks that need to be detected to establish watermark presence

cal value suggested by Equation 6.2. Overall, 9 different fractions, 10 different attack levels, and 5 different tolerance levels were introduced and watermark was detected in 42167 out of 46045 watermarked files with a cumulative probability of 91.5%.

6. ANALYSIS

We shall now analyze the capacity and security properties of the watermarking scheme as compared to previous schemes such as [1, 13].

6.1 Capacity

In our scheme, γ tuples out of every 100 tuples are selected for watermarking. Thus the capacity of our scheme is given by $C = \frac{\eta}{\gamma}$ where η is the total number of tuples. The capacity is, theoretically, same as capacity of previous scheme ([1]. Distortion levels Ξ used in our schemes are much higher. The modified values can later be reversed back to original values upon purchase of full version of the data set. Allowing higher distortion results in more attributes selected for marking actually getting marked thereby increasing the capacity in practice.

6.2 Security

In terms of security, the possible attacks to consider are given in Section 3. Next we discuss the security of our proposed solution.

A1: Random bitwise flipping attack. If we assume that the attacker has complete knowledge of Ξ and υ. The attacker can now

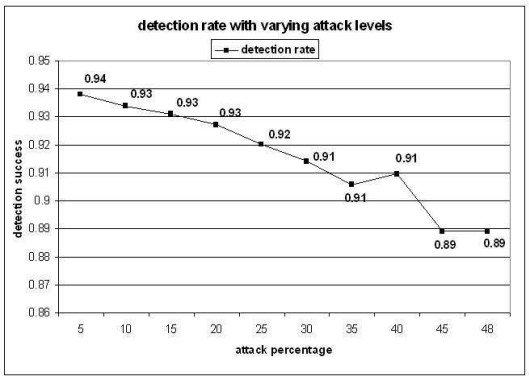

Figure 3: Effect of changing attack levels on detection

choose randomly tuples ζ and flip all the ξ_i LSBs of attribute A_i $(1 \leq i \leq \upsilon)$ in those tuples. This attack is successful if the attacker can toggle sufficient marked bits such that detection algorithm detects less than τ watermarked bits correctly. Hence the attacks succeeds only when attacker modifies at least $\omega - \tau + 1$ watermarked bits where $\omega = \frac{\eta}{\gamma}$ is the total number of tuples marked. The probability of this attack is given by Equation 6.2 [1]. This probability is the same as [1, 13]. For $\gamma = 50$, the worst case scenario is when attacker changes 48% of the tuples and the success probability of attack is merely 11% as confirmed by experiments and shown in Figure 3. If the attacker changes more than half the tuples, a) the usability would be assumed to be severely affected, and, b) watermark would be detected in the bitwise complemented relation.

$$P(A) = \sum_{i=\tilde{\tau}}^{\omega} \frac{\binom{\omega}{i}\binom{\eta-\omega}{\zeta-i}}{\binom{\eta}{\zeta}} \tag{5}$$

A2: Subtractive attack. This type of attack is similar to the previous attack in that the attacker has to again remove at least $\omega - \tau + 1$ marked tuples out of η tuples such that the detection algorithm detects less than τ matches. The probability of this attack is same as the previous attack (random bitwise flipping attack).

A3: Sorting

If an attacker re-sorts the tuples based on any attribute, it does not effect the detection algorithm. Since the watermark detection is carried out of each tuple independently, any change in order does not effect the outcome of the detection algorithm. Sorting attack was given significant importance while deciding difference expansion method to be used.

A4: Secondary watermarking

Let us consider a situation where Alice watermarks relation R resulting in relation R_a ($R \xrightarrow{ins(Alice)} R_a$) and distributes it for trial. The attacker Mallory modifies R_a to R'_a and re-watermarks R'_a resulting in relation R_m. R'_a still contains Alice's watermark with a high probability p and Alice's watermark is successfully removed by Mallory with a probability $1-p$ (According to experimental results, $p \approx 0.89$ for $\gamma = 50$). R_m contains Mallory's watermark with probability 1 since it has not been modified after watermark insertion. Let R_a accidently contain Mallory's watermark with a negligible probability δ ($\delta \approx 0$).

The judge asks Alice and Mallory to run detection algorithm on R_a and R_m respectively. Both Mallory's and Alice's watermarks are successfully detected in their respective relations. Mallory's restored relation is R'_a and Alice's restored relation is R. With a high probability p, Alice's watermark is detected in R'_a but Mallory's watermark is detected with an extremely low probability δ in R. Thus it becomes evident that Mallory inserted the watermark in the relation already watermarked by Alice and thereby Alice is the rightful owner. In this way, the current watermarking scheme defeats secondary watermarking attacks.

7. CONCLUSION

In this paper, we have proposed a reversible and blind database watermarking model. The maximum distortion introduced to the attributes is limited to the tolerance parameter Ξ. It is, in practice, desirable to have distortion on the higher side since the watermarking is reversible. The distorted database is available to everyone and the accurate database can be purchased upon payment by users by reversing the watermarking. The proposed scheme is successful in achieving the major objective of eliminating the shortcomings of irreversible schemes like [1] mentioned in Section 1. The capacity of the proposed watermarking scheme is high and the attack resistance probability between 89 and 98 percent. Our future research is directed towards increasing the watermark carrying capacity and level of attack resistance in a reversible and blind watermarking model.

8. REFERENCES

[1] R. Agrawal and J. Kiernan. Watermarking relational databases. In *Proceedings of the 28th International Conference on Very Large Databases VLDB*, 2002.

[2] Rakesh Agrawal, Peter J. Haas, and Jerry Kiernan. Watermarking relational data: framework, algorithms and analysis. *The VLDB Journal*, 12(2):157–169, 2003.

[3] A.M. Alattar. Reversible watermark using the difference expansion of a generalized integer transform. *IEEE Transactions on Image Processing*, 13(8):1147–1156, 2004.

[4] A. Bors and I. Pitas. Image watermarking using dct domain constraints. In *Proceedings of IEEE International Conference on Image Processing (ICIP'96)*, volume III, pages 231–234, September 1996.

[5] Gordon W. Braudaway. Protecting publicly-available images with an invisible image watermark. In *Proceedings of IEEE International Conference on Image Processing (ICIP'97)*, Santa Barbara, California, October 1997.

[6] M.U. Celik, G. Sharma, M.A. Tekalp, and E. Saber. Reversible data hiding. In *Proceedings of International Conference on Image Processing*, volume 2, pages 157–160, September 2002.

[7] Chin-Chen Chang, Wei-Liang Tai, and Min-Hui Lin. A reversible data hiding scheme with modified side match vector quantization. In *AINA '05: Proceedings of the 19th International Conference on Advanced Information Networking and Applications*, pages 947–952, Washington, DC, USA, 2005. IEEE Computer Society.

[8] I.J. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. Technical Report 128, NEC Research Institute, August 1995.

[9] Ingemar Cox, Joe Kilian, Tom Leighton, and Talal Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, 1997.

[10] Ingemar J. Cox, Joe Killian, Tom Leighton, and Talal Shamoon. Secure spread spectrum watermarking for images, audio, and video. In *IEEE International Conference on Image Processing (ICIP'96)*, volume III, pages 243–246, 1996.

[11] David Gross-Amblard. Query-preserving watermarking of relational databases and xml documents. In *Proceedings of the 20th ACM Symposium on Principles of Database Systems*, pages 191–201, June 2003.

[12] Fei Guo, Jianmin Wang, and Deyi Li. Fingerprinting relational databases. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 487–492, New York, NY, USA, 2006. ACM Press.

[13] Gaurav Gupta and Josef Pieprzyk. Reversible and semi-blind relational database watermarking. In *Proceedings of International Conference on Signal Processing and Multimedia Applications*, July 2007.

[14] Yingjiu Li and Robert Huijie Deng. Publicly verifiable ownership protection for relational databases. In *Proceedings of the ACM Symposium on Information, computer and communications security*, pages 78–89, New York, NY, USA, 2006. ACM Press.

[15] Yingjiu Li, Huiping Guo, and Sushil Jajodia. Tamper detection and localization for categorical data using fragile watermarks. In *DRM '04: Proceedings of the 4th ACM workshop on Digital rights management*, pages 73–82, New York, NY, USA, 2004. ACM Press.

[16] Radu Sion, Mikhail Atallah, and Sunil Prabhakar. Rights protection for relational data. *IEEE Transactions on Knowledge and Data Engineering*, 16(12):1509–1525, December 2004.

[17] Jun Tian. Reversible data embedding using a difference expansion. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(8):890–896, 2003.

[18] Yong Zhang, Xia-Mu Niu, and Dongning Zhao. A method of protecting relational databases copyright with cloud watermark. *Transactions of Engineering, Computing and Technology*, 3:170–174, 2004.

[19] Yong Zhang, Bian Yang, and Xia-Mu Niu. Reversible watermarking for relational database authentication. *Journal of Computers*, 17(2):59–66, 2006.

[20] Z.H. Zhang, X.M. Jin, J.M. Wang, and D.Y. Li. Watermarking relational database using image. In *Proceedings of 3rd International Conference on Machine Learning and Cybernetics*, volume 3, pages 1739–1744, August 2004.