Identification and Proof of Ownership by Watermarking Relational Databases

Vidhi Khanduja, Member, IACSIT and O. P. Verma

Abstract—Rapid increase in copying and distributing digital assets are major concerns to content owners. In this paper, we propose a new robust secure and imperceptible embedding mechanism to resolve the two important concerns namely, owner identification and proof of ownership. The steps of proposed mechanism for watermarking relational databases mainly involves encoding and decoding on numerical attribute of relational database in three phases: 1) Watermark preparator, 2) Watermark position detector and 3) Watermark Embedder or Detector. The first phase identifies owner as owner’s identity is used to get watermark bits. In second phase position, watermarks are to be embedded are identified using secret key and pseudorandom generators. This phase marks multiple attributes with varying number of candidate bit positions within a single tuple. In the third phase watermarks are embedded in Encoder. While decoder extracts watermarks and detects database piracy.

Index Terms—Relational Database, Watermark, Copyright protection, Ownership identification, Proof of ownership.

I. INTRODUCTION

Internet is an excellent distribution system for digital media because it is inexpensive, eliminates warehousing and stock and delivery is almost instantaneous. Copying and distributing digital assets have become layman’s task. However, owners of such digital assets are concerned about the copyright of their products.

The general solution to this problem is watermarking. A watermark is information that can be used for ownership verification and proof of identity of owner of digital products. Watermarking techniques allows owner of data to embed an imperceptible watermark into data which can include anything the owner chooses. Watermark embedding for relational data is made possible by the fact that real data can tolerate a small amount of error without any significant degradation in their usability [1]. There are many application contexts for which data represent an important asset, ownership of which must be carefully enforced. Any watermarking system should satisfy following properties:

1. Embedding effectiveness: The probability that the embedder will successfully embed a watermark in a randomly selected database.
2. Fidelity: The perceptual quality of watermarked content.
3. Blind detection: Detecting watermark should not require original database.
4. Robustness: The ability of watermark to survive normal processing of content
5. Security: The ability of the watermark to resist hostile attacks.
6. Modification and multiple watermarks: The possibility of changing embedded watermarks or embedding several watermarks in one tuple of the database.

In the proposed method all above properties are taken in to the consideration.

II. RELATED WORK


The watermarking algorithm for relational databases proposed in [1] assume that database relations can be watermarked in some attributes, such that changes in few values do not affect their applications. This algorithm embeds watermarks only in one attribute out of several candidates attributes in a tuple.

In this paper we proposed a technique to securely and randomly select any number of attributes out of selected candidate attributes for embedding watermarks in varying number of least significant bits. We have devised a secure and imperceptible embedding mechanism that provides not only proof of ownership but also owner identification.

III. PROPOSED ALGORITHM

Proposed watermarking system consists of two subsystems watermark encoder and respective decoder.

Watermark Encoder: It embeds desired watermarks into relational database. This task is achieved using three steps as shown in Fig.1.
A. Watermark Preparator

Watermark to be inserted is selected by owner of the database. The watermark must be chosen such that it reflects owner’s identity. This step identifies the identity of database’s copyright holder as watermark. Thus ensures owners’ identification.

Owner selects the watermarking text ‘W’ and secret key ‘K1’ to create a watermark to be embedded.

The algorithm
1. Input the values of ‘W’ and K1
2. For each character Cᵢ in W do
   3. \(W_b[i] = C_i + K1\)
   [end of for loop]
4. \(W_p = \text{binary}(W_b)\) // binary\((W_b)\) function converts number to binary.

Line 2 in the algorithm indicates that owner chosen text is read character by character and addition of each character with secret key is computed in line 3 to give integer value. These values are stored in \(W_b\) array. At line 4, binary of \(W_b\) is taken and finally stored in \(W_p\) array.

B. Watermarking Position Detector

Suppose R is relation whose scheme is \(R (P, A_0, A_1, ..., A_{n-1})\) where P is primary key attribute and R contains total \(n\) attributes. Let owner selects ‘\(v\)’ number of numeric attributes that are candidates for marking. Each attribute \(A_i\) is numeric with values such that small changes in \(LBA_{Ai}\) least significant bits are imperceptible. We consider that each attribute has varying number of candidate bit positions i.e. \(LBA_{Ai}\). The gap \(\gamma\) [1] is a control parameter that determines the number \(w\) of tuples marked out of total \(r\) tuples via approximate relationship \(w = r / \gamma\). The t.X represents the value of attribute X in tuple t ∈ R.

In this algorithm cryptographic pseudorandom sequence generators (CPSG) [12] are used that generates computationally infeasible sequence of numbers which depends on initial seed. Pseudorandom generators generate same fixed sequence of numbers every time if fixed initial seed is given.

The following functions are used in the algorithm
1) MAC: For each tuple \(t'\) in relation R, secure Message Authentication Code[13] is computed using secret key K2 known only to owner of the database and tuple’s primary key t.P.
2) Next(CPSG1): This generates next number in random sequence using CPSG1.
3) Selectattr(next(CPSG1)): An another pseudorandom sequence generator CPSG2 is created with initial seed as next(CPSG1) whose output is a vector with number of states equivalent to \(v\). These states decide what all attributes in a tuple are selected for watermark. Since output of this depends on previous pseudorandom generator, this increases the level of security.

For erasing a watermark, the attacker needs to correctly guess the tuples that are marked and the selected attributes with their corresponding selected bit positions.

The algorithm
1. Input the value of secret key K2.
2. For each tuple \(t \in R\) do
3. Compute \(\text{MAC} = H(K2|| t.P || K2)\)
   Where, \(H()\) is secure hash function, and ‘\(||\)’ is concatenation operator.
4. Seed CPSG1 with MAC of each tuple.
5. If \((\text{next}(CPSG1) \mod \gamma) = 0\) then
   //mark the tuple
6. \(\text{Attrindc}[\ ] = \text{selectattr}(\text{next}(CPSG1))\)
7. For each value in \(\text{Attrindc[}\ ]\) \n8. If (\(\text{Attrindc[i]}\) equals 1) // mark the attribute
9. Select \(A_{i}\) for marking
10. Bitindex j = \(\text{next}(CPSG1) \mod LBA_{Ai}\)
    // mark corresponding bit position
    [end of if at line 8]
    [end of for loop at line 7]
    [end of if at line 5]
    [end of for loop at line 2]

C. Embed Watermark

For selected attribute \(A_{i}\) and corresponding selected bit position \(j\), we embed watermark generated \(W_p\) in relational database R. If number of watermark bits in \(W_p\) are less than number of detected watermarked positions in step2 we repeat the watermark bits in \(W_p\) again.

Watermark Decoder:

Fig. 2 shows watermark decoder which detects whether the database is pirated or not.

In detection process, the first two steps of watermark insertion are followed. Once attribute indices and bit positions are found in marked database \(S\) using secret key K2, we test whether or not the bits value matches the values that should have been assigned by insertion algorithm and count the number of matches \(\text{matchcnt}(m)\) against total number of watermarks \(\text{totalcount}(w)\). If there are very many matches or very few matches we suspect piracy [1]. We fix small value \(\alpha \in (0, 1)\) and sets
\[
\tau = \max \{ t \mid 0, \frac{w}{2} \}; \sum_{t=1}^{w-1} b \left( i; w, \frac{j}{2} \right) \geq 1 - \alpha \} \tag{1}
\]

where

\[
b(i; n, p) = np!(1 - p)^{n-i}
\]

We suspect piracy if either \( m < \tau \) or \( m > w - \tau \), as probability of so few or so many matches under null hypothesis is less than or equal to \( \alpha \). \( \alpha \) is called significance level of the test.

**Functions used in watermark detector algorithm:**

1. **Match(s, Ai, j):** This function tests whether or not the bit value of attribute \( s.Ai \) at position \( j \) matches the values that is assigned by embedding algorithm i.e \( Wp \) and returns 1 if match found.
2. **Threshold(totalcount, \( \alpha \)):** This function calculates threshold value \( \tau \) using (1). Total number of watermarks inserted and value of \( \alpha \) are passed to the function.

**The algorithm**

// **Watermark Preparation**
1. Input the values of watermark information \( W \) and secret key K1
2. For each character \( C_i \) in \( W \) do
   - \( W[p[i]] = C_i + K1 \)
3. \( \text{WP} = \text{binary}(W) \) // binary(W) function converts number to binary.

// **Watermark Position Detection**
5. Input the value of secret key K2.
6. Totalcount = matchcnt = 0
7. For each tuple \( t \in S \) do
8.    Compute \( \text{MAC} = H(K2 || t.P || K2) \)
     where, \( H() \) is secure hash function, and \( \text{||} \) is concatenation operator.
9.    Seed CPSG1 with MAC of each tuple.
10. If (next(CPSG1) mod \( \gamma \) equals 0) then
     // mark the tuple
11. Attribnde[ ] = selectattr(next(CPSG1))
12. For each value in Attribnde[ ]
13.    If (Attribnde[i] equals 1) // mark the attribute
14.    Select \( A_i \) for marking
15. Bitindex \( j = \text{next(CPSG1)} \mod LBA_i \)
     // mark corresponding bit position
16. totalcount = totalcount + 1

// **Watermark Detector**
17. matchcnt = matchcnt + match(s.Ai, j)
18. \( \tau = \text{threshold(totalcount, } \alpha) \)
19. If ((matchcnt < \( \tau \)) or (matchcnt > totalcount - \( \tau \))) then
20.    Suspect piracy
21. [end of if at line 19]
22. [end of if at line 13]
23. [end of for loop at line 12]
24. [end of if at line 10]
25. [end of for loop of line 7]

Detecting watermark is blind technique as it does not require original database and watermarks can be detected even in small subset of watermark relations as long as sample contains some of the marks (discussed in Section IV).

For ownership identity, the watermark bits are extracted from database \( S \) and reverse of the watermark preparation algorithm is followed to get repeated watermarked text from which original \( W \) is extracted.

### III. EXPERIMENTS AND ANALYSIS

The proposed algorithm is tested and evaluated on an experimental database consisting of approximately 10000 tuples. We ran the experiment on MATLAB environment and found that our algorithm is robust against following types of attacks.

#### A. Subset Deletion Attack

In this, attacker may delete randomly selected subset of tuples of watermarked database so that watermark will be removed.

We performed the experiment by deleting selected subsets of database and watermark extracted was recorded as shown in Fig. 3. Our experiment revealed that even if 90% of subsets are deleted approximately 12% of watermarks are still detected. Thus it still provides as a proof of ownership and to great extent ownership identification as watermarking bits are repeatedly embedded, we can extract meaningful information by further processing.

![Fig. 3. Watermark Detection in Subset deletion attack.](image)

#### B. Subset Addition attack

![Fig. 4. Watermark Detection in Subset addition attack.](image)
Ownership identification information is extracted little impact on extraction of watermarked database. In Fig.4. Our experiment revealed that this attack has very little impact on extraction of watermarked database. Ownership identification information is extracted completely.

IV. CONCLUSION

Owner identification and proof of ownership issues are resolved in this paper. This paper proposes a secure robust and imperceptible algorithm. We divide embedding algorithm in three phases: Watermark preparator, watermark position detector and watermark embedder. The ownership identification issue is resolved by embedding owner’s identity as watermark in Preparator phase. Position detector phase securely identifies multiple attributes with varying number of candidate bit positions of the single table. Embedder inserts watermarks at identified bit positions of multiple attributes of relational database. The robustness of the proposed algorithm was verified against number of database attacks.

REFERENCES


Om Prakash Verma received his B.E. degree in Electronics and Communication Engineering from Malaviya National Institute of Technology, Jaipur, India in 1991 and M. Tech. degree in Communication and Radar Engineering from Indian Institute of Technology (IIT), Delhi, India, in 1996 and PhD(S) from University of Delhi, Delhi, India in 2011. From 1992 to 1998 he was assistant professor in Department of Electronics & Communication Engineering, at Malaviya National Institute of Technology, Jaipur, India. He joined Department of Electronics & Communication Engineering, Delhi College of Engineering (now Delhi Technological University) Delhi, India, as Associate Professor in 1998. Currently, he is Head of Department of Information Technology at Delhi Technological University, Delhi. He is also the author of more than 15 publications in both international journal and conference proceedings. He has guided more than 15 M. Tech. student for their thesis. He has authored a book on Digital Signal Processing in 2003. He is a Principal investigator of an Information Security Education Awareness project, sponsored by Department of Information Technology, Government of India. His research interests include image processing, application of fuzzy logic in image processing, application of evolutionary algorithm in image processing, artificial intelligent and digital signal processing.