Privacy and Ownership Preserving of Outsourced Medical Data

Elisa Bertino
Purdue University
bertino@cs.purdue.edu

Beng Chin Ooi
National University of Singapore
ooibc@comp.nus.edu.sg

Yanjiang Yang, Robert H. Deng
Institute for Infocom Research
yanjiang;deng@i2r.a-star.edu.sg

Abstract

The demand for the secondary use of medical data is increasing steadily to allow for the provision of better quality health care. Two important issues pertaining to this sharing of data have to be addressed: one is the privacy protection for individuals referred to in the data; the other is copyright protection over the data. In this paper, we present a unified framework that seamlessly combines techniques of binning and digital watermarking to attain the dual goals of privacy and copyright protection. Our binning method is built upon an earlier approach of generalization and suppression by allowing a broader concept of generalization. To ensure data usefulness, we propose constraining Binning by usage metrics that define maximal allowable information loss, and the metrics can be enforced off-line. Our watermarking algorithm watermarks the binned data in a hierarchical manner by leveraging on the very nature of the data. The method is resilient to the generalization attack that is specific to the binned data, as well as other attacks intended to destroy the inserted mark. We prove that watermarking could not adversely interfere with binning, and implemented the framework. Experiments were conducted, and the results show the robustness of the proposed framework.

1. Introduction

Nowadays, effective sharing of medical data is essential to foster the collaboration within the health care community and with other parties such as research institutes, pharmaceutical and insurance companies, so as to enhance the quality and efficacy of health care provision. For example, a hospital may need to outsource clinical records in its autonomous databases to a research institute in an attempt to discover a new drug or evaluate a new therapy. Such need is clearly shown by research trends in the area of health care management and procedures that are increasingly based on extensive analysis of medical data. The dissemination of medical data could also be to satisfy legal requirements. As reported by the National Association of Health Data Organization in 1996, 37 states in the United States had legislative mandates to gather personal health information from hospitals for cost-analysis purposes [22].

The direct release of medical data invariably violates individual privacy. Data must be thus properly processed before delivery in order to protect the privacy of the individuals they refer to. A straightforward method for achieving individual privacy is to de-identify (anonymize) the data, by replacing any explicit identifying information by some random placeholders. For instance, a randomized value may be used to substitute the name or social security number of each patient. This alone, however, does not suffice to guarantee the full anonymity of medical data as pointed out by numerous studies (see for example, [13, 28, 26, 29]). An example often outlined is re-identification by linking attributes such as birth date, zip code that are shared by the anonymized medical data and some externally collected voting records. This has motivated many more advanced approaches in the literature (see Section 2). Of particular interest is the approach of *generalization and suppression* [26, 28, 29] that represents values by corresponding more general but semantically accordant alternatives.

The sharing of medical data also exposes data holders to the threat of data theft. Related to this, yet another important protection requirement regarding outsourced medical data arises, that is, how to protect data ownership (copyright). It is quite obvious that medical data are an important asset to the data holders who have collected and compiled the information. Incentives to unauthorized data distribution arise from an increasingly thriving *data industry* where firms such as biotech companies collect, compile, share or sell (bio)medical data for profits. Even though there are laws concerning copyright and ownership rights, we need effective mechanisms to establish and protect the holders' rightful possession of the data. Consequently and naturally, digital watermarking techniques, initially proposed for the protection of multimedia content [6, 15], have been recently also applied to relational data. As such, digital watermarking techniques represent a viable solution for the problem of enforcing ownership of medical data. However, a main difference of medical data with respect to data from different domains is represented by the need of also assuring privacy. It is thus clear that when dealing with outsourced medical data, both individual privacy and data ownership must be protected. To meet these dual needs, we propose a framework that integrates techniques of binning and digital watermarking. Under our framework, the medical data to be

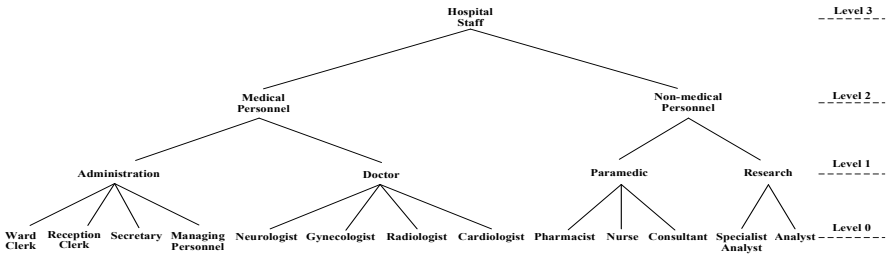


Figure 1. A domain hierarchy tree (DHT) for a column representing the types of person roles in a medical domain

outsourced would undergo two consecutive steps of binning and watermarking, respectively. The main contributions of our work include:

1. A unified framework that seamlessly combines binning and digital watermarking for the protection of both individual privacy and data ownership. We give both theoretical and experimental analysis on the "seamless-ness" of the combination.

2. A binning algorithm that enforces the functionality of "binning". The method bins downward, and extends an earlier approach of generalization and suppression by allowing a broader concept of generalization.

3. A hierarchical watermarking scheme that is resilient to various attacks attempting to remove the embedded mark, and especially robust against the newly discovered *generalization attack*. In addition, we propose an elegant solution to the rightful ownership problem concerning watermarking.

4. The adoption of usage metrics for preserving data quality with respect to the intended usage. We define our usage metrics by modeling information loss, and propose an off-line enforcement of usage metrics.

5. Experimental studies of the proposed framework.

Compared to existing approaches, a main innovative aspect of our work is represented by a downward binning process to address the satisfaction of k-anonymity specification, due to the off-line enforcement of usage metrics; our watermarking algorithm is a novel hierarchical scheme that exploits the very nature of the underlying data, which also provides a neat solution to the rightful ownership problem.

Organization: We review related work in Section 2. In Section 3, we give an overview of our framework. We then proceed to detail our binning algorithm and watermarking algorithm in Sections 4 and 5, respectively. In Section 6, we present a theoretical analysis on the seamlessness of our framework. Section 7 provides experimental results and Section 8 concludes the paper.

2. Background and Related Work

Two classes of techniques closely related to our work are information disclosure control and relational database watermarking, and we shall review them in this section. Given a relational table containing medical data, columns can be categorized into three types based on the identifying information each contains. Columns that explicitly identify individuals (e.g., social security number) are known as identifying columns, and columns containing potentially identifying information that could be linked with other data sets to re-identify individuals are called quasi-identifying columns. Typical examples of quasi-identifying columns include zip code, birth date, etc. The other columns contain no identifying information. In this paper, we restrict ourselves to quasi-identifying columns unless explicitly stated otherwise.

Information Disclosure Control

Information disclosure arises when either the identity of an individual is directly revealed or something about an individual can be derived from the released data. By convention, we call the former *identity disclosure* and the latter *attribute disclosure* [18]. We only discuss the identity disclosure problem in this paper, and refer interested readers to [31] for in-depth discussions on the attribute disclosure.

One well known approach to identity disclosure control is to transform quasi-identifying columns to entertain *k*-anonymity constraint (k is a constant), i.e., data are generalized and suppressed in such a way that every record is indistinguishable from at least *k*-1 other records, so that no search can be narrowed down to a particular individual [13, 26, 28, 29]. The satisfaction to *k*-anonymity can also be understood as: records containing the same value constitute a bin, and the size of every bin is at least equal to k. By definition, generalization deals with replacing a value with a more general but semantically accordant value, while suppression deals with preventing data releases. Generalization of categorical attributes is based on the fact that the representation of medical data can be normally arranged into a *domain hierarchy tree* (DHT), where the most general description of the data is at the root of the tree while the leaves denote the most specific descriptions. Figure 1 shows a DHT on the type of roles: leaf nodes represent all possible particular roles a column may assume, and generality of the description increases with the level along the tree, until the root node that distinguishes no specificity. A generalization proceeds by replacing the values represented by the leaf nodes by their corresponding ancestor nodes at a higher level. A

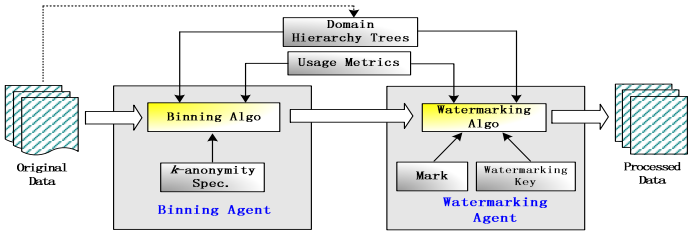


Figure 2. Protection framework for outsourced medical data

valid generalization in [26, 28, 29] requires all its generalization nodes be at the same level in the domain hierarchy tree.

Clearly, generalization and suppression result in a loss of specificity, thereby making the re-identification process harder. However, the tradeoff between the level of privacy and the amount of information loss must be carefully evaluated, as too much generalization could possibly render the data useless while slight generalization could not provide adequate protection. [14] suggested associating usage based metrics with the process of meeting k-anonymity. Our framework incorporates the same idea of usage metrics, but we define a different set of metrics, and more importantly, our metrics can be enforced off-line. Metrics in [14] are defined in accordance with the broader notion of generalization allowed therein, which does not require all generalization nodes stay at the same level. The *binning* method in [19] follows a similar broader definition of generalization. Considering the flexibility and finer granularity it offers, our binning algorithm also includes such a broader notion in extending the generalization and suppression in [26, 28, 29]. Moreover, the off-line enforcement of usage metrics enables a downward binning in our context, which has efficiency advantage over binning that proceeds upwards.

Another approach to the identity disclosure problem is to perturb the data by adding noise or swapping values, while at the same time maintaining some statistical properties of the entire data set [17, 11]. It is again vital to determine the right tradeoff between information loss and privacy – a topic which is now under active research [8, 32]. Other approaches dealing with data privacy and confidentiality but addressing issues different from ours include [12, 30, 1, 3, 21, 4, 5].

Watermarking of Relational Data

Digital watermarking has long been investigated for copyright protection, mainly over multimedia content, e.g., images and video clips [6, 15]. There have been recent efforts in watermarking relational databases. Due to the very nature of relational data, watermarking techniques for databases turned out not to be a direct deployment of techniques for multimedia data. A seminal approach to watermarking relational data is presented in [2]. However, the use of Least Significant Bits (LSB) embedding in the scheme makes it inherently vulnerable, as a simple flipping of LSBs would completely destroy the inserted mark. [24] proposed a method for watermarking numbers that is robust because the mark embedding relies on data distribution rather than on trivial LSB modification. The idea has later been integrated in a framework for watermarking numeric attributes of relational databases [25]. A theoretical investigation on watermarking techniques for databases and XML documents is presented in [7], which attempts to achieve watermarking while preserving a set of parametric queries in a specified language.

Another approach [23] was recently proposed dealing with watermarking categorical attributes in databases. In essence, the data to be watermarked in our context become categorical after binning, so our watermarking also reduces to handling categorical data. Unfortunately, such approach cannot be directly applied to our case because it is susceptible to a kind of *generalization attack* (see Section 5).

3. Overview of Our Framework

To simultaneously attain the goals of protecting individual privacy and copyright protection regarding outsourced medical data, we combine techniques of binning and digital watermarking into a unified framework. As shown in Figure 2, the framework comprises two key components, i.e., binning agent and watermarking agent, dedicated to binning and watermarking, respectively. In the framework, the medical data to be outsourced would undergo two consecutive steps of transformation. Specifically, the binning agent first bins the data to satisfy k-anonymity specification. Afterwards, the binned data are watermarked by the watermarking agent by inserting within the data a mark, which, upon extraction, asserts provable ownership. The data resulting from these transformations are then expected to adequately protect both privacy and copyright, thereby qualified for outsourcing. Both binning and watermarking are governed by usage metrics in order to preserve data usability. Next, we shall discuss some specific aspects of the framework.

Usage Metrics: Usage metrics define a set of maximal distortions that binning and watermarking are allowed to introduce with respect to the intended data usage (see Section 4). Transformation exceeding the bounds is assumed to render the data useless.

k-anonymity Specification: k-anonymity specification in-

cludes the system parameter k, and possibly also the set of quasi-identifying columns to be binned and other relevant constraints pertaining to binning.

Binning Agent: Driven by the binning algorithm, the binning agent attempts to bin the data to satisfy k-anonymity specification while at the same time adhering to the usage metrics. After binning, each bin is guaranteed to contain at least k records, so no specific individual can be identified. The binning algorithm takes as input the original data, the *k*-anonymity specification, the domain hierarchy trees for each quasi-identifying attribute, and the usage metrics. We suggest a preprocessing step to create the domain hierarchy trees and determine the system parameters.

Watermarking Agent: The watermarking agent continues to process the binned data by embedding an owner-specific mark. The underlying watermarking algorithm exploits a secret watermarking key (may contain several elements), known only to the data owner, to manipulate the process of mark embedding. Without having possession of the secret watermarking key, no one can erase the inserted mark from the data. Watermarking also observes usage metrics, ensuring that it does not corrupt the data in terms of the anticipated usage; the domain hierarchy trees are needed as well for inspection by our watermarking algorithm.

4. Binning Algorithm

Our binning algorithm extends the approach of generalization and suppression in [26, 28, 29] by allowing a broader notion of generalization as in [14], which does not require all generalization nodes of a generalization to be necessarily at the same level of the domain hierarchy tree. In particular, a valid generalization G is represented by a set of generalization nodes S_G in the domain hierarchy tree that satisfy the following condition: *The path from every leaf to the root along the tree encounters one (to guarantee generalizability) and only one (to guarantee deterministic generalization) generalization node in* S_G. This definition includes the case of a leaf node itself being a generalization node. We have seen domain of a categorical attribute being organized into a domain hierarchy tree; we next describe the generalization of a *numeric column*. It is accomplished by first dividing the domain space of the column into a series of disjoint intervals, and then pairwise combining them into a binary tree. With the tree, generalization proceeds in the same way as for a categorical attribute. As an example, Figure 3 depicts the construction of a binary domain hierarchy tree for the column Age with domain [0, 150). In order to avoid over-binning the data, intervals should be of *moderate* size (smaller) and they need not to be of equal size.

Clearly, binning makes data less specific (more general), thereby resulting in some information loss. It would make no sense to meet k-anonymity specification if that renders the data useless, thus data quality must be preserved. We

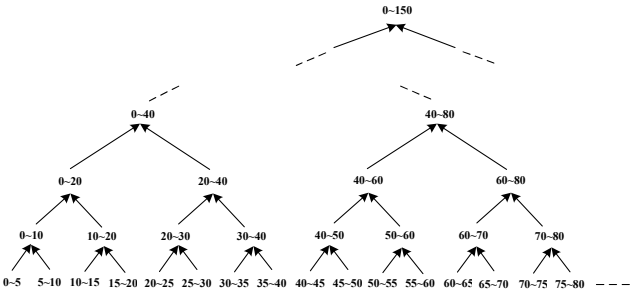


Figure 3. Constructing binary DHT for a numeric attribute

suggest constraining the binning process to abide by *usage metrics* specifying a set of maximal allowable information loss. More information loss than as specified would substantially degrade the data quality with respect to the intended data usage.

4.1 Usage Metrics

Consider first a categorical column c that associates with a domain a hierarchy tree T, e.g., Figure 1. If *Pharmacist* is generalized to *Paramedic*, under our definition of generalization, child nodes of *Paramedic* would become indiscriminatable. This in turn implies that all entries in c containing *Pharmacist/Nurse/Consultant* would become indiscriminatable. This concept of indiscrimination leads to our approach for quantifying information loss $InfLoss_c$ for the column c as follows. Suppose a generalization results in a set of generalization nodes $\{p_1, p_2, ..., p_M\}$; let S_i be a set containing the leaf nodes of the subtree that is rooted at p_i, and the number of entries in c containing values in S_i be n_i, $i = 1..M$. Information loss $InfLoss_c$ is defined as

$$InfLoss_c = \frac{\sum_{i=1}^{M}(n_i \frac{|S_i|-1}{|S|})}{\sum_{i=1}^{M} n_i} \quad (1)$$

where $S = S_1 \bigcup S_2 \bigcup ... \bigcup S_M$ is the set of leaf nodes of the tree T. We allow some leaf nodes to remain ungeneralized given that k-anonymity specification is already met, in which case $|S_i| = 1$.

We next consider a numeric attribute c, e.g., Age. Suppose the domain of c, whose lower and upper bounds are L and U, respectively, is generalized into M intervals. The lower and the upper bounds for these intervals are L_i and U_i, respectively, $i = 1..M$. Let n_i be the number of entries in the column c whose values fall between L_i and U_i, $InfLoss_c$ is then defined as

$$InfLoss_c = \frac{\sum_{i=1}^{M}(n_i \frac{U_i-L_i}{U-L})}{\sum_{i=1}^{M} n_i} \quad (2)$$

Once all $InfLoss_i$, $i = 1..CN$ (CN is the total number of the generalized columns) are determined, a normalized

loss $InfLoss$ is computed by averaging over all generalized columns in the table:

$$InfLoss = \frac{\sum_{i=1}^{CN} InfLoss_i}{CN} \quad (3)$$

Likewise, other forms of information loss, e.g., total information loss can be defined. In general, the usage metrics for controlling information loss are defined as following:

$$InfLoss_i \leq bd_i \ \ \forall i = 1, ..., CN$$
$$InfLoss \leq bd_{avg} \quad (4)$$

where $\mathcal{B} = \{bd_1, ..., bd_{CN}\} \subset \mathcal{R}$ and $bd_{avg} \in \mathcal{R}$ define the bounds for maximal allowable information loss.

In practice, the enforcement of the above metrics in a normal way might not be ideal as it involves calculating information loss and in turn checking against the bounds after every step of binning. Fortunately, we can implement an off-line enforcement, yielding a set of *maximal generalization nodes* in each domain hierarchy tree. Maximal generalization nodes are defined as 1) constituting a valid generalization; 2) each being the highest node in the domain hierarchy tree to which the corresponding leaf nodes can be generalized under the usage metrics. Usage metrics in the form of maximal generalization nodes are obviously much easier to enforce, only requiring that none of the leaf nodes be generalized beyond its corresponding maximal generalization node. It is preferable that the maximal generalization nodes are directly given as the usage metrics, rather than being transformed from the form of Equation (4).

We note that a generalization comprising the maximal generalization nodes trivially satisfies k-anonymity specification given that the data are *binnable*. The point is to meet k-anonymity while minimizing information loss. It is thus clear that binning would yield a set of generalization nodes that are lower than or at most equal to the maximal generalization nodes. This reasonably reflects the underlying principle that binning is not allowed to damage data usage. Let us consider the earlier example of generalizing a numeric attribute, where we suppose the set of intervals in satisfying k-anonymity is depicted by the leaf nodes of the tree in Figure 4: enforcement of the usage metrics might most likely allows for further generalizations, yielding the set of maximal generalization nodes denoted as elliptic nodes.

4.2 Binning

We decompose binning into two steps, i.e., mono-attribute binning and multi-attribute binning. The mono-attribute binning step bins attributes individually so that each transformed attribute satisfies k-anonymity. The multi-attribute binning step is required because, while each attribute satisfies k-anonymity, combinations of them may not.

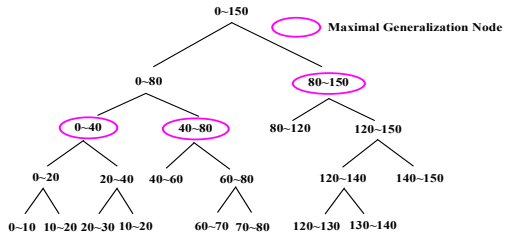


Figure 4. A DHT by enforcing usage metrics

Consider an example of a transformed table, where 36 people have an age between $25 \sim 50$ and 8 people are doctors, each satisfying k-anonymity specification with $k = 6$. However, there might be only 4 people who are aged between $25 \sim 50$ who are also doctors.

For ease of referencing, we list in Table 1 the variables and functions that will be used in this and the next section.

Notation	Meaning
tr	the domain hierarchy tree for an attribute
tbl	the table to be protected
$mingends$	the set of minimal generalization nodes
$maxgends$	the set of maximal generalization nodes
$ultigends$	the set of ultimate generalization nodes
k	the system parameter for k-anonymity
k_1, k_2, η	elements of the secret watermarking key
wm, wmd	actual and replicated mark, respectively
Parent(nd, tr)	returns the parent node of nd in tr
Children(nd, tr)	returns the set of child nodes of nd in tr
Siblings(nd, tr)	returns nd together with its sibling nodes in tr
Leaves(tr)	returns the set of leaf nodes of tr
SubTree(nd, tr)	returns the subtree of tr rooted at nd
Duplicate(wm)	duplicates wm to produce wmd
Val2Nd(v, nds[])	returns the node in nds[] that represents v
Nd2Val(nd)	returns the value represented by nd
SetμBit(v, b)	sets the least significant bit of v to be the bit b
Index(nd, S)	returns the index of nd in the set S
MajorVot(wmd)	majority voting over wmd

Table 1. Variables and Functions

4.2.1 Mono-attribute Binning

For an individual attribute, our binning starts from the maximal generalization nodes downwards along the domain hierarchy tree, until reaching a set of lowest nodes that constitute a valid generalization catering to k-anonymity specification. We term such nodes *minimal generalization nodes*. Our way of downward binning is an advantage offered by the off-line enforcement of usage metrics. The mono-attribute binning is basically an *exhaustive* trial procedure in a search for the minimal generalization nodes. For this reason, compared to previous work that bins upward along the tree (e.g., [19]), downward binning turns out to be more efficient. The intuition is that the higher level on the tree, the less nodes are to be tried. Note that the observance of usage metrics is di-

rectly accomplished by starting binning from the maximal generalization nodes. Figure 5 outlines the algorithm for generating the set of minimal generalization nodes.

GenMinNd(tr, $maxgends$, tbl, k)
1. $mingends \leftarrow$ NULL
2. **foreach** node $nd \in maxgends$
3. $subtr \leftarrow$ SubTree(nd, tr)
4. $mingends \leftarrow mingends \bigcup$ SubGMN($subtr$, tbl, k)

SubGMN(tree str, tbl, k)
1. **if** NumTuple(str, tbl) $< k$
2. **return** NULL
3. **forany** node $nd \in$ Children(str.root, tr)
4. **if** NumTuple(SubTree(nd, str), tbl) $< k$
5. **return** $\{str$.root$\}$
6. $tmpset \leftarrow$ NULL
7. **foreach** $nd \in$Children(str.root, str)
8. $subtr \leftarrow$ SubTree(nd, str)
9. $tmpset \leftarrow tmpset \bigcup$ SubGMN($subtr$, tbl, k)
10. **return** $tmpset$

NumTuple(tree str, tbl)
1. int $num = 0$
2. **foreach** tuple $t_i \in tbl$
3. **if** t_i.val $\in$ Leaves(str)
4. $num \leftarrow num + 1$
5. **return** num

Figure 5. Mono-attribute binning algorithm

We employ a simple rationale in generating a minimal generalization node: a node is *minimal* if itself meets k-anonymity, but not all of its child nodes do. This might lead to an over-generalization of the data. A more aggressive strategy could be capitalized on, e.g., a node is not minimal if *any* of its child nodes satisfies k-anonymity.

4.2.2 Multi-attribute Binning

Multi-attribute binning involves further binning attributes, each of which already satisfies k-anonymity. However, for an individual attribute, the set of allowable generalizations for the purpose of multi-attribute binning is already defined by the nodes between the minimal generalization nodes and the maximal generalization nodes. Consider Figure 6: the set of allowable generalizations constrained by the minimal generalization nodes and the maximal generalization nodes are enumerated as {30, 31, 45, 46, 33, 22}, {30, 31, 32, 33, 22}, {30, 31, 21, 22}, {20, 45, 46, 33, 22}, {20, 32, 33, 22} and {20, 21, 22}. As a result, the set of allowable generalizations for the entire table is the enumeration of different combinations of allowable generalizations for all attributes. Let the number of quasi-identifying columns be CN, and

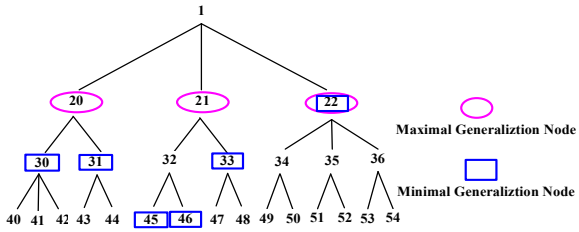


Figure 6. A DHT for illustrating multi-attribute binning

n_i be the number of allowable generalizations for column i, then the total number of allowable generalizations for the table is $\prod_{i=1}^{CN} n_i$.

Among these allowable generalizations, some do not satisfy k-anonymity, and are thereby invalid; the remaining are valid for k-anonymity. Nevertheless, not all these valid generalizations are equally satisfactory. The point here is to choose among them an *ultimate generalization* that results in the minimal information loss. Nodes in this ultimate generalization are called *ultimate generalization nodes*. Clearly, the calculation of information loss can be done by using Equation (1), (2) and (3), although this may not be ideal as it may incur unacceptable computation penalty. Instead, we prefer simplifying this calculation by solely considering "specificity loss" regarding the domain hierarchy trees. Let the total number of leaf nodes of a tree be N and the number of generalization nodes of an allowable generalization be N_g, we define specificity loss due to generalization to be $(N - N_g)/N$. This approach of estimating specificity loss results in a more efficient implementation, but it may reduce accuracy.

Figure 7 outlines the above approach for determining the ultimate generalization nodes. The function EnumGen(.) enumerates all distinct combinations of allowable generalizations among attributes, and the function Selection(.) determines the generalization that incurs least specificity loss.

GenUltiNd($mingends$[1..CN], $maxgends$[1..CN], tr[1..CN])
1. **for** i = 1..CN
2. $allowblgens[i] \leftarrow \{gen_j$ | gen$_j$ is a generalization constrained by $mingends[i]$, $maxgends[i]$ in $tr[i]\}$
3. $allgens \leftarrow$ EnumGen($allowblgens[i]$, $i = 1..$CN)
4. $validgens \leftarrow \{$gen$_j$ | gen$_j \in allgens \bigwedge$ gen$_j$ satisfies k-anonymity$\}$
5. $ultigen \leftarrow$ Selection($validgens$)

Figure 7. Multi-attribute binning algorithm

4.2.3 Binning Algorithm

A relevant observation to make is that the identifying columns are most likely to be the key attributes (e.g., primary key) of the table, containing the most important part of information. Hence it is frequently useful to maintain the

identifying columns traceable to the data holder in health care domain. For instance, as reported in [9], in some cases patients may benefit from being traced in research such as the assessment of treatment safety. Moreover, many real-world clinical projects such as those in [16] and in [10] support traceability of the medical data. Based on this observation, our binning algorithm adopts an one-to-one replacement for data in the identifying columns. In particular, we replace each data by its encrypted value that is generated by an encryption function $\mathcal{E}()$ e.g., DES or AES. We point out that keeping the identifying columns unsuppressed and unmanipulated further is also important for watermarking. Figure 8 outlines our complete binning algorithm, comprising the encryption of the identifying columns and the binning of the quasi-identifying columns. Given the ultimate generalization *ultigen* yielded by multi-attribute binning, the function Bin(.) works by simply replacing each value in the quasi-identifying columns by the value represented by its corresponding node in *ultigen*.

Binning(tbl, $ultigen$)
1. **foreach** tuple $t_i \in tbl$
2. t_i.ident.val $\leftarrow \mathcal{E}(t_i$.ident.val)
3. t_i.quasi-ident.val $\leftarrow$ Bin(t_i.quasi-ident.val, $ultigen$)

Figure 8. Binning algorithm

5. Watermarking Algorithm

By its very nature, watermarking modifies the data to be watermarked, thereby further degrading data quality. Watermarking works under a general assumption that the underlying data can tolerate a certain degree of quality degradation. The tolerance closely relates to the bandwidth for insertion, implying that watermarking would fail unless the data can be modified. The discovery of the available bandwidth appears to be challenging in the case of watermarking relational data [25, 23]. We next explain how to find the desired bandwidth channel for insertion in the binned data.

5.1 Bandwidth Channel

In our context, columns of a table after binning become essentially categorical, and data modification by watermarking is equivalent to the permutation of data. We advocate that a binned table can actually accommodate some degree of data permutation, thereby providing the desired bandwidth channel for watermarking.

From earlier discussions, we know that generalization of a node in the hierarchy tree to its parent node renders indiscrimination among this node and its sibling nodes. In essence, a random permutation of values represented by these nodes equals the effect of the generalization. As long as such a generalization is allowed, watermarking relying on the data permutation would definitely work. Recall that the set of maximal generalization nodes defined by usage metrics are normally atop the set of ultimate generalization nodes resulting from binning. Hence, generalizations between the two levels still respect usage metrics, which in turn guarantee the viability of watermarking. It is important to notice a special case where a ultimate generalization node itself is also a maximal generalization node. Permutation of such nodes might result in information loss above the threshold set by usage metrics. However, watermarking affects only a small fraction of the data set, and hence such excessive loss is expected to be minor. As a matter of fact, this is the price that any watermarking must pay. More importantly, we can readily tackle this scenario by slightly modifying the way a maximal generalization node is defined. Specifically, in determining the set of maximal generalization nodes, the bounds in Equation (4) are given slightly lower than actually required for sustaining data usage, so that a small fraction of the table is allowed to be generalized to the values represented by the maximal generalization nodes. Note however that such transformation on a large scale would definitely destroy the data.

5.2 Watermarking at A Single Level

A direct way to take advantage of the above bandwidth channel is to consider permutation at the level of each ultimate generalization node (together with its sibling nodes). The exact primitive enabling bit insertion works as follows. Suppose an ultimate generalization node p needs to be permutated, and p and its sibling nodes compose a sorted set S. To insert a bit b, our basic idea for determining a target node q in S such that $p \rightarrow q$ encodes the bit b is: the index of q in S is even, if $b = 0$; the index of q in S is odd, if $b = 1$. However, this does not suffice since some elements in S may not be ultimate generalization nodes, so if the target node q is not an ultimate generalization node, validity of the generalization (see Section 4) is violated. To solve this issue, we shall continue the permutation process downward among the child nodes of q, and possibly even lower, until an ultimate generalization node is reached. Our definition of *generalization* guarantees the reachability. This idea of achieving embedding by data permutation is similar to [23], but we do within finer domains (sub-domain of the column), and more importantly we have solid justifications for permutation. Unfortunately, watermarking at this single level is susceptible to a kind of *generalization attack* that can completely destroy the inserted bits without knowing the watermarking key.

Generalization attack

The generalization attack is specific to the binned data. It works as follows: the attacker starts a further generalization on the watermarked table, generalizing each value to the value represented by a higher generalization node in the domain hierarchy tree. Because of the gap between the

maximal generalization nodes and the ultimate generalization nodes, the table would sustain data usage. The generalization attack appears fatal as it does not require the secret watermarking key at all. A careful analysis indicates that it is the way we consider watermarking only at the level of ultimate generalization nodes that makes possible the attack. To thwart this attack, we must additionally watermark all intermediate levels between the maximal generalization nodes and the ultimate generalization nodes. This constitutes the basic idea of our hierarchical watermarking scheme.

5.3 A Hierarchical Watermarking Scheme

In the hierarchical watermarking, we consider watermarking at every level, from the maximal generalization nodes to the ultimate generalization nodes. Specifically, for an ultimate generalization node p to be permutated, watermarking starts by first determining the maximal generalization node q that corresponds to p, followed by executing permutations downward along the domain hierarchy tree from the level of the child nodes of q, until the target node is an ultimate generalization node. The exact primitive enabling permutation at each level is the same as above. Consider Figure 6 for example (for illustration's sake, we need to intentionally take the minimal generalization nodes therein as the ultimate generalization nodes), where node 46 is going to be permutated. First, the corresponding maximal generalization node 21 is determined. Next, permutation proceeds within nodes 32 and 33. If the target node is node 33, then permutation stops; otherwise, the permutation continues within nodes 45 and 46, and eventually stops.

To avoid a large scale alteration, watermarking is ideally restricted to a (small) portion of the whole data set. We leverage on the (encrypted) identifying columns of the binned table to select some tuples for embedding, recalling that the encrypted identifying columns are assumed to keep intact[1]. Based on a secret key k_1 together with a secret tunable parameter η, tuples t_i in the table tbl satisfying the following equation are chosen for insertion:

$$\mathcal{H}(t_i.\text{ident},\ k_1) \bmod \eta = 0 \quad \forall t_i \in tbl \tag{5}$$

where $\mathcal{H}()$ is a cryptographic hash function e.g., MD5 or SHA1, and tbl.ident denotes the encrypted identifying columns of tbl. Note that the way of secretely selecting tuples directly pertains to the resilience of watermarking.

Typically, the available bandwidth is greater than the bit length $|wm|$ of the mark wm. This affords a multiple embedding of wm for robustness reasons. That is, we repeatedly embed wm many times until the available bandwidth is exhausted. In mark detection phase, the final mark is determined by *majority voting* over all the recovered copies.

[1]In case the identifying columns cannot be relied on, we can establish virtual key attributes as in [20] by turning to other columns

A straightforward way to achieve multiple embedding is to duplicate wm for l times into wmd, as long as we attempt an l-embedding, and then to insert wmd in place of wm.

Take tbl.c, a quasi-identifying column of tbl for example, our hierarchical watermarking algorithm by integrating the above ideas, is outlined in Figure 9. The function MaxGNd(nd, tr, $maxgends$) returns the maximal generalization node that associates with nd.

Embedding(tbl, tr, $maxgends$, $ultigends$, k_1, k_2, η, wm)
1. bits $wmd \leftarrow$ Duplicate(wm)
2. **foreach** tuple $t_i \in tbl$
3. **if** $\mathcal{H}(t_i.\text{ident}, k_1) \bmod \eta = 0$
4. node $targnd \leftarrow$ Val2Nd(t_i.c, $ultigends$)
5. $targnd \leftarrow$ MaxGNd($targnd$, tr, $maxgends$)
6. **do**
7. $targnd \leftarrow$ Permutate($targnd$, tr, t_i, k_1, k_2, wmd)
8. **while** $targnd \notin ultigends$
9. t_i.c $\leftarrow$ Nd2Val($targnd$)

Permutate(node nd, tr, tuple t_i, k_2, bits wmd)
1. sortedset $S \leftarrow \{s_i \mid s_i \in \text{Children}(nd, tr)\}$
2. int $indx \leftarrow \mathcal{H}(t_i.\text{ident}, k_2) \bmod |S|$
3. $indx \leftarrow$ SetμBit(indx, $wmd[\mathcal{H}(t_i.\text{ident}, k_2) \bmod |wmd|]$)
4. **return** s_{indx}

Detection(tbl, tr, $maxgends$, $ultigends$, k_1, k_2, η, wm)
1. bits $wmd \leftarrow$ NULL /* set wmd to be empty */
2. **foreach** tuple $t_i \in tbl$
3. **if** $\mathcal{H}(t_i.\text{ident}, k_1) \bmod \eta = 0$
4. node $tmpnd \leftarrow$ Val2Nd(t_i.c, $ultigends$)
5. bit[] b = NULL, int i = 0 /* reset */
6. **do**
7. sortedset $S \leftarrow \{s_i \mid s_i \in \text{Siblings}(tmpnd, tr)\}$
8. int $indx \leftarrow$ Index($tmpnd$, S)
9. $b[i] \leftarrow indx$&1
10. $i \leftarrow i + 1$
11. $tmpdnd \leftarrow$ Parent($tmpnd$, tr)
12. **while** $tmpnd \notin maxgends$
13. $wmd[\mathcal{H}(t_i.\text{ident}, k_2) \bmod |wmd|] \leftarrow$ MajorVot(b)
14. $wm \leftarrow$ MajorVot(wmd)

Figure 9. Hierarchical watermarking algorithm

In the algorithm, we exploit distinct keys k_1 and k_2 for different calculations, which is vital in ensuring that there is no mutual correlation between these calculations. Notice that the hierarchical scheme enables to insert several copies of a bit at every single embedding position, and the actual number is equal to the number of levels from the corresponding maximal generalization node to the ultimate generalization node. Thus, when recovering a bit from a single embedding position, the bit is determined by majority voting. Interestingly, in the voting process, we can assign a different

weight to each copy from a distinct level, depending on its credit in determining the bit. This is of special use when enforcing the policy that the copy from a higher level is more reliable than that from a lower level.

5.4 Resolving Rightful Ownership Problem

Robustness to attacks attempting to erase the embedded mark is among the fundamental requirements of a sound watermarking. However, this does not necessarily imply its sufficiency in establishing ownership, because of the attacking scenarios in Figure 10 (D_x, W_x and K_x are respectively the original data, the mark and the secret watermarking key of the entity x, D_w and $\overline{D}_w$ denote the watermarked data).

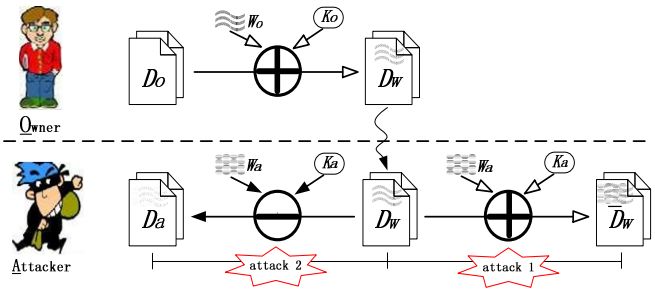


Figure 10. Rightful ownership attacks

Attack 1: the attacker inserts his bogus mark W_a into D_w, which is the owner's valid watermarked data, to create his bogus $\overline{D}_w$. Now that both W_o and W_a are contained in $\overline{D}_w$, the attacker and the owner can both claim the ownership over $\overline{D}_w$. This attack can be resolved by requiring the attacker and the owner each to present his original data. As the attacker's "original" data D_w contains W_o of the owner, false ownership claim by the attacker is clear.

Attack 2: In this case however, the attacker "extracts" W_a from D_w to obtain his bogus original data D_a, so that $D_a \oplus_{k_a} W_a = D_w$, where $\oplus_{k_a}$ denotes the embedding function under key k_a. This attack is more subtle to handle, since it does not always hold that D_a contains W_o and D_o does not contain W_a. So far, the only practical solution in multimedia watermarking is to restrict W_o to be $\mathcal{F}(D_o)$, where $\mathcal{F}(.)$ is an one-way function, so that given D_w, it is impossible to acquire D_a satisfying $\mathcal{F}(D_a) = W_a$ by the attacker.

These attacks are in fact the rightful ownership problem originally raised in [27] in multimedia context. It will be of particular interest to see how the rightful ownership problem is handled in our case. We notice that virtually none of the existing proposals for watermarking databases has provided a satisfactory solution to this problem, as either they considered merely one case of it (e.g., [2, 20]) or they did not address it at all (e.g., [25, 23]). Results from the multimedia sector show that without invoking a third party for certifying the watermarked data D_w, the rightful ownership problem is solvable only when the original data are available in court. We believe this directly applies to the context of databases. Considering the large number of data a table contains, we actually suspect the practicality of presenting to the judge the entire original table as court proof in other proposals. Surprisingly, the nature of the binned data enables us to elegantly resolve this problem in our context. Recall that the identifying columns of a binned table to be watermarked are in encrypted format, which means the attacker has no way to know the clear-text. So the mark in our scheme is specified by applying the one-way function $\mathcal{F}(.)$ to a certain statistical value v (e.g., mean) of these clear-text of the identifying columns. In resolving ownership dispute, the owner presents v; decrypts the identifying columns and does the same statistical computation over the decrypted data to get v'; compares the two as valid if $|v - v'| < \tau$, where τ is a predefined threshold; extracts the mark from the table in dispute and compares it with $\mathcal{F}(v)$ as usual in a normal watermarking scheme. Note that most probably, the watermarked table in dispute had been attacked, e.g., some tuples were deleted or some spurious tuples were added, and this explains why we acquire the mark from a statistical value instead of the actual clear-text.

The proposed solution is specific to our integration of binning and watermarking, since a normal database does not have such encrypted attributes. In nature, we do not violate "original data as court proof", whereas the integrated property of our framework provides an effective means to get over direct reliance on the entire original table.

6. Analysis

We next explore the seamlessness of our framework from a theoretical perspective. In other words, we are concerned with the effect watermarking has on the result of binning. The main issue is related to the fact that watermarking in our context involves permutation such that some tuples in a bin may be permutated to other bins, and thus some bins may have, after watermarking, a size less than k. This means that watermarking may compromise the satisfaction to k-anonymity of binning. Without loss of generality, we restrict our discussions to a particular quasi-identifying column c, which corresponds to a domain hierarchy tree having m maximal generalization nodes N_i ($i = 1..m$), and n_i ultimate generalization nodes associated with each node N_i. We further make the following assumptions: (i) bins that correspond to the ultimate generalization nodes are of equal size; (ii) when a bit-embedding proceeds downward from N_i, all the n_i ultimate generalization nodes associated with N_i have equal probability of becoming the target node when permutations halt. The actual effect of watermarking on binning can be reduced to the way any particular bin (*BIN*) that corresponds to a ultimate generalization node *UGN* is affected by any bit-embedding (*E*).

Lemma 1. *Let the maximal generalization node corresponding to UGN be N_k, and the probability of E reducing the bin size of BIN by* 1 *be Pr$^-$, then Pr$^-$ = $\frac{n_k-1}{n_k \sum_{i=1}^{k} n_i}$.*

Proof: Intuitively, for E to reduce the bin size of BIN by 1, it must hold that as per our hierarchical watermarking algorithm, 1) the bit chosen by E for insertion comes from BIN; 2) afterwards, E executes downward permutations (starting from N_k) among the n_k ultimate generalization nodes that correspond to N_k, and the target node of such permutations is not UGN. From assumption (i), probability that the tuple chosen by E comes from BIN is $\frac{1}{\sum_{i=1}^{m} n_i}$, and from assumption (ii), probability of the target node not being UGN is $\frac{n_k-1}{n_k}$. Hence, altogether $Pr^- = \frac{1}{\sum_{i=1}^{m} n_i} \times \frac{n_k-1}{n_k} = \frac{n_k-1}{n_k \sum_{i=1}^{k} n_i}$. ♢

Lemma 1 states the probability of any particular bit-embedding E permutating a tuple out of a particular bin BIN. We next check the probability of E permutating a tuple from another bin to BIN.

Lemma 2. *Let the maximal generalization node corresponding to UGN be N_k, and the probability of E increasing the bin size of BIN by* 1 *be Pr$^+$, then Pr$^+$ = $\frac{n_k-1}{n_k \sum_{i=1}^{k} n_i}$.*

Proof: For E to increase the bin size of BIN by 1, it must hold that 1) E selects the tuple for insertion from any, but UGN, of the n_k ultimate generalization nodes that are associated with N_k ; 2) the target node of the downward permutations is UGN. From assumption (i), probability of the former is $\frac{n_k-1}{\sum_{i=1}^{m} n_i}$, and from assumption (ii), probability of the latter is $\frac{1}{n_k}$. Hence, $Pr^+ = \frac{n_k-1}{\sum_{i=1}^{m} n_i} \times \frac{1}{n_k} = \frac{n_k-1}{n_k \sum_{i=1}^{k} n_i}$.♢

Lemma 1 and Lemma 2 suggest that on average, the watermarking process would neither decrease nor increase the bin size of any bin since $Pr^- = Pr^+$. We therefore conclude that watermarking does not interfere with binning in the satisfaction of k-anonymity specification under the two ideal assumptions.

It is of importance to examine the assumptions from a practical perspective. Making valid the first assumption is not that hard: we can incorporate "restrained swapping" (e.g., swapping tuples among bins that correspond to sibling nodes) into binning. In contrast, the second assumption is more tricky, because its validity totally rests with the locality of ultimate generalization nodes on the domain hierarchy tree. Even so, we believe that by relaxing the two assumptions, watermarking still cannot seriously interfere with binning because: 1) only a small percentage of the whole data gets watermarked; 2) and the use of hash function in the "suitability" selection step (Equation (5)) renders a uniform culling, which means no particular bin will be drastically affected. To attest this, we have done experiments and obtained consistent results (see next section). After all, we have a simple yet practical method to tackle the interference by applying $k+\epsilon$ (ϵ is a small number) to binning in meeting k-anonymity specification. A conservative method for determining ϵ would be as follows: let s be the biggest bin size and S be the sum of all bin sizes, then $\epsilon = (s/S) * |wmd|$.

7. Experimental Studies

We implemented and conducted extensive experiments on the above algorithms. The real world data set we experimented on include one (randomized) identifying column and five quasi-identifying columns, whose schema is $\mathcal{R}$(*ssn*, *age*, *zip_code*, *doctor*, *symptom*, *prescription*). By a preprocessing step, we created a DHT for each quasi-identifying column: the DHT for *symptom* is based on the International Classification of Diseases (ICD-9), and other attributes are on self-defined ontology, e.g., that for *age* is similar to Figure 3 but of narrower intervals. The whole data set contains around 20000 tuples. Experiments were done on a PC with 2G CPU and 512M RAM, and source codes were written in Microsoft C++. A main *simplification* we made is that a set of maximal generalization nodes is directly given to each column as usage metrics.

7.1 Robustness of Binning

First, our experiments focus on testing the binning algorithm in satisfying k-anonymity. By providing to the algorithm different values of k, we recorded the corresponding loss of information. Figure 11 shows the relationship of k versus information loss.

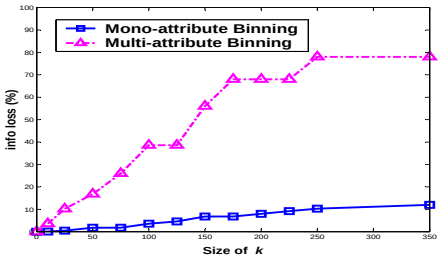
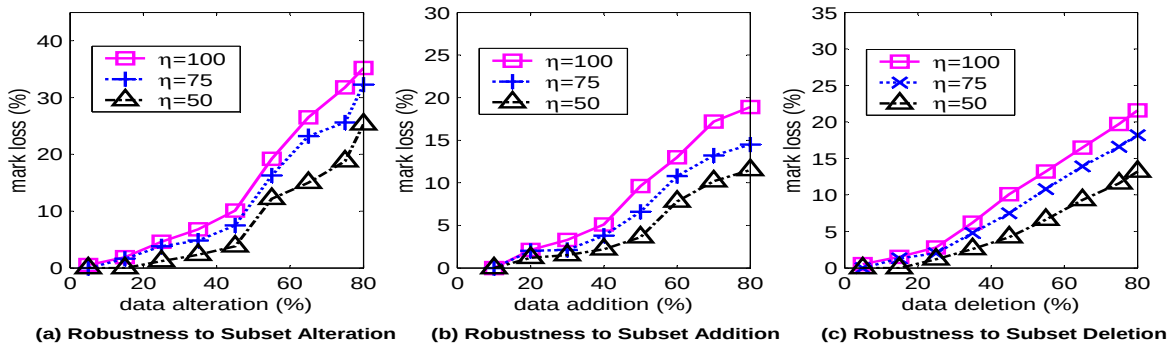


Figure 11. k vs. information loss

From the figure, multi-attribute binning causes much more information loss than mono-attribute binning, and once k increases to a certain extent, information loss reaches a saturation point and becomes rather stable. This is consistent with the rationale in determining a valid minimal generalization node (Section 4.2), and this could be further optimized if the more aggressive strategy as introduced there is employed. Further, we should also note that information loss is closely related to the data size, the number of quasi-identifying columns and k.



(a) Robustness to Subset Alteration (b) Robustness to Subset Addition (c) Robustness to Subset Deletion

Figure 12. Robustness of hierarchical watermarking

7.2 Robustness of Watermarking

In this set of experiments, we test the robustness of the hierarchical watermarking scheme to the attacks that endeavor to destroy the embedded mark, while in the absence of the secret watermarking key. The following experiments were conducted by implementing a multiple embedding of a 20-bit mark.

- *Subset Alteration*

In these attacks, the attacker chooses at random a subset of the data and then modifies them arbitrarily without affecting the rest of the data. We vary the size of the randomly altered data, and calculate the corresponding mark loss. Figure 12 (a) outlines the results. Clearly, the results show that our watermarking scheme performs well against this attack. Even in the case of more than 70% of data loss, our scheme loses only approximately 30% of mark bits. Another fact shown in the figure is that smaller η (more bandwidth) offers more resilience, whereas more alteration to the data would be incurred. This is a trade-off that must be carefully considered in practice.

- *Subset Addition*

In these attacks, new tuples are frequently added to the watermarked set by the malicious attacker. Although this attack does not involve erasing existing bits, it nevertheless misleads the selection criteria (Equation (5)) to falsely take some of the newly-added tuples as watermarked, thereby introducing errors in majority voting the final mark. Keep in mind that if the size of the new data exceeds the original data size, priority of the former would dominate the latter. Figure 12 (b) highlights the scheme's robustness to the Subset Addition attacks. The results reflect the fact that the newly-added bogus bits do not take precedence over the existing bits in the majority-voting process.

- *Subset Deletion*

The attacker randomly deletes a percentage of the tuples in an attempt to remove the mark. To test the effect of dropping tuples to the loss of mark bits, we continually delete some tuples each time by the following SQL clause:

DELETE FROM $\mathcal{R}$ WHERE SSN $>$ lval$_i$ AND SSN $<$ uval$_i$

where lval$_i$ and uval$_i$ define bounds of the i^{th} deletion, within which the tuples are to be deleted. Figure 12 (c) plots the series of mark loss due to the deletions. From the figure, it indicates that the hierarchical scheme is resilient to the Subset Deletion attacks, and mark loss increases almost linearly with the amount of data deleted.

We also tested the information loss due to watermarking, and Figure 13 presents the results. Clearly, information loss caused by watermarking is minor.

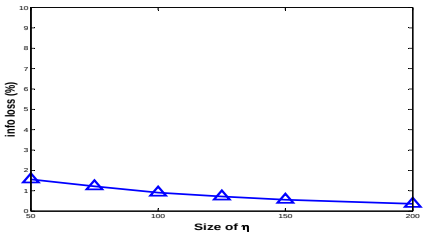


Figure 13. Information loss of watermarking

7.3 Seamlessness of Framework

Finally, we shall examine how watermarking interferes with binning, complementing the theoretic analysis in the preceding section. The results are presented in Figure 14, where the data in each column respectively represents the total number of bins, number of bins having bin size changed and number of bins having bin size less than k. It can been seen that a majority of the bins are affected by watermarking, whereas the interference is minor in terms of satisfying k-anonymity: none of the bins cannot meet k-anonymity after watermarking. This is consistent with our analysis that watermarking does not dramatically affect binning in its compliance with k-anonymity specification.

8. Conclusion

Two important issues inherent to the outsourcing of medical data are the protection of individual privacy and copyright protection over the data. To meet these dual needs,

k \ Attribute	*age*	*zip_code*	*doctor*	*symptom*	*prescription*
10	73 / 58 / 0	96 / 82 / 0	20 / 18 / 0	56 / 53 / 0	97 / 86 / 0
20	68 / 61 / 0	88 / 79 / 0	20 / 17 / 0	52 / 48 / 0	90 / 82 / 0
45	52 / 48 / 0	81 / 72 / 0	20 / 17 / 0	47 / 38 / 0	79 / 71 / 0
100	42 / 35 / 0	62 / 56 / 0	18 / 15 / 0	36 / 31 / 0	59 / 48 / 0

Total number of bins / Number of bins having binsize changed / Number of bins having binsize < k

Figure 14. Effect of watermarking on binning

we have integrated techniques of binning and digital watermarking into a unified framework, so as to provide comprehensive protection for outsourced data. Under our framework, medical data are in turn binned to meet k-anonymity specification, and watermarked to provide copyright protection. We have discussed at length the development of the binning algorithm and the watermarking algorithm that provide the two core functions in our framework, and developed an elegant solution to the rightful ownership problem regarding watermarking, which may be difficult to solve in the context of other approaches. From both theoretical and practical perspectives, we proved that watermarking would not substantially interfere with binning in the satisfaction to k-anonymity. Experimental results showed the robustness of the proposed framework.

References

[1] R. Agrawal, A. Evfimievski, R. Srikant, *Information Sharing Across Private Databases*, Proc. SIGMOD, pp. 86 - 97, 2003.

[2] R. Agrawal, J. Kiernan, *Watermarking Relational Databases*, Proc. VLDB, VLDB, pp.155-166, 2002.

[3] R. Agrawal, R. Srikant, *Privacy-preserving data mining*, Proc. SIGMOD, pp. 439-450, 2000.

[4] M. Bawa, R. J. Bayardo Jr., R. Agrawal, *Privacy-Preserving Indexing of Documents on the Network*, Proc. VLDB, pp.922-933, 2003.

[5] L. Bouganim, P. Pucheral, *Chip-Secured Data Access: Confidential Data on Untrusted Servers*, Proc. VLDB, pp. 131-142, 2002.

[6] I. Cox, J. Boom, and M. Miller, *Digital Watermarking*, Morgan Kaufmann, 2001.

[7] D. G. Amblard, *Query-preserving Watermarking of Relational Databases and XML documents*. Proc. PODS, pp. 191-201, 2003.

[8] J. D. Ferrer, J. M. Sanz, and V. Torra, *Comparing SDC Methods for Microdata on the Basis of Information Loss and Disclosure Risk*, Proc. of NTTS and ETK, 2001.

[9] Food and Drugs Administration, *Medwatch: The FDA Safety Information and Adverse Event Reporting Program*, http://www.fda.gov/medwatch/.

[10] T. A. Ferris, G. M Garrison, H. J. Lowe, *A Proposed Key Escrow System for Secure Patient Information Disclosure in Biomedical Research Databases*, Proc. AMIA Annual Symposium, pp. 245-249, 2002.

[11] S. E. Fienberg, *Statistical Perspectives on Confidentiality and Data Access in Public Health*, Stat Med, 20(9-10), pp. 1347-1356, 2001.

[12] H. Hacigms, B. R. Iyer, C. Li, S. Mehrotra, *Executing SQL Over Encrypted Data in the Database-service-provider Model*. Proc. SIGMOD, pp. 216-227, 2002.

[13] A. Hundepool, L. Willenborg, *μ- and τ- argus: Software for Statistical Disclosure Control*. Proc. 3^{rd} International Semiar on Statistical Confidentiality, 1996.

[14] V. S. Iyengar, *Transforming Data to Satisfy Privacy Constraints*, Proc. SIGKDD, pp.279-288, 2002.

[15] N. F. Johnson, Z. Duric, and S. Jajodia, *Information Hidding: Steganography and Watermarking - Attacks and Countermeasures*, Kluwer Academic Publishers, 2000.

[16] D. Kalra, P. Singleton, D. Ingram, J. Milan, J. MacKay, D. Detmer, A. L. Rector, *Security and Confidentiality Approach for the Clinical E-Science Framework (CLEF)*, Proc 2^{nd} UK E-Science "All Hands Meetings", pp825-832, 2003

[17] J. Kim and W. Winkler, *Masking Microdata Files*, ASA (American Statistical Association) Proc. on Survey Research Methods, pp. 114-119, 1995.

[18] D. Lambert, *Measures of Disclosure Risk and Harm*, Journal of Official Statistics, 9(2), pp. 313-331, 1993.

[19] Z. Lin, M. Hewett, and R. B. Altman, *Using Binning to Maintain Confidentiality of Medical Data*, American Medical Informatics Association Annual Symposium, pp. 454-459, 2002.

[20] Y. J. Li, V. Swarup, and S. Jajodia, *Constructing a Virtual Primary Key for Fingerprinting Relational Data*, Proc. ACM Workshop on Digital Rights Management, pp. 133-141, 2003.

[21] G. Miklau, D. Suciu, *Controlling Access to Published Data Using Cryptography*, Proc. VLDB, pp. 898-909, 2003.

[22] National Assoc. Health Data Organizations, *A Guide to State-Level Ambulatory Care Data Collection Activities*, 1996.

[23] R. Sion, *Proving Ownership over Categorical Data*, Proc. ICDE, 2004.

[24] R. Sion, M. Atallab, and S. Prabhakar, *On Watermarking Numeric Sets*. Proc. IWDW, LNCS 2613, pp. 130-146, 2002.

[25] R. Sion, M. Atallah, and S. Prabhakar, *Rights Protection for Relational Data*, Proc. SIGMOD 2003, 98-109.

[26] L. Sweeney, *Datafly: A System for Providing Anonymity in Medical Data*, Proc. Database Security, pp. 356-381, 1998.

[27] S. Craver, N. Memon, B. Yeo, and M. Yeung, *Can Invisible Watermarks Resolve Rightful Ownerships?* Technique Report RC 20509, IBM Research Division, 1996.

[28] P. Samarati, *Protecting Respondents' Identities in Microdata Release*, IEEE Trans. Knowledge Engineering, 13(6), pp. 1010-1027, 2001.

[29] P. Samarati, L. Sweeney, *Protecting Privacy when Disclosing Information: K-Anonymity and Its Enforcement Through Generalization and Suppression*, Technical Report, SRI International, 1998.

[30] S. Schleimer, D. S. Wilkerson, A. Aiken, *Winnowing: Local Algorithms for Document Fingerprinting*, Proc. SIGMOD, pp. 76 - 85, 2003.

[31] L. Willenborg and T. D. Waal, *Statistical Disclosure Control in Practice*, Lecture Notes in Statistics, Vol. 111, Springer-Verlag, 1996.

[32] W. Yancey, W. Winkler, R. Creecy, *Disclore Risk Assessment in Perturbative Microdata Protection*, Technical Report 2002-01, Statistical Research Division, Bureau of the Census.