The dbMark: A Benchmarking System for Watermarking Methods for Relational Databases

Stavros Kyriakopoulos, Theodoros Tzouramanis
Department of Information and Communication Systems Engineering, University of the Aegean, Samos, Greece
stavrosicsd@gmail.com, ttzouram@aegean.gr

Yannis Manolopoulos
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
manolopo@csd.auth.gr

***Abstract*—Digital technology keeps falling prey to the ease with which the unauthorized reproduction and distribution of digital objects can be achieved. Research on copyright protection of digital data must counterbalance the failures of legal measures against digital piracy. Digital watermarking tops the list of technical countermeasures and recent research has focused on proposing watermarking methods for relational data as well as on the types of attacks which aim at removing or destroying the watermark. With the database field of research in urgent need of developing tools and platforms for the benchmarking of these watermarking methods, the present work has undertaken to construct a benchmarking system for the automatic evaluation and fair comparison of watermarking schemes for relational data. The new software tool is called dbMark and its construction has taken into consideration the characteristics of a large set of parameters across a wide range of database watermarking applications. This article presents and comments upon the results of several experiments that evaluate the performance of three different watermarking methods taking into consideration several parameters and types of attacks. There are indications that the dbMark can be used effectively as the primary platform for the experimental evaluation of watermarking methods for relational data. The source code and the executable file of the dbMark are provided free-of-charge on the Web.**

Keywords—Database as a digital asset; database security; database copyright protection; digital watermarking; database watermarking methods; benchmarking; open-source software tool.

I. INTRODUCTION

The business world requires that databases be delivered as services products [1], which puts significant pressure on data providers to create web services that allow users to search and access databases remotely. Further, huge data warehouses with measurements that can be used for data mining purposes are in great demand. On the other hand, providers are unwilling to allow their data to remain potentially exposed to the predatory interest of malicious users and they require, consequently, mechanisms that are geared to provide not only the protection of intellectual rights but also to make it possible to carry out integrity and authenticity verification, the localization of unauthorized data alteration and the tracing of threatening entities. As the typical answer to the increasing demand for such requirements, *digital watermarking* emerged by providing the process of hiding digital information in a carrier such as a database. When this process is used for copyright protection, it is called *robust* watermarking; if it is used for integrity or authenticity verification and localization, it is called *fragile* watermarking; and if it is used for tracing an attacker it is called *fingerprinting*.

Watermarking is a very popular and effective mechanism for protecting multimedia assets. In digital images, on the other hand, this technology has been in use for over twenty years [2] and a great number of methods have been proposed since then. In digital audio and video, watermarking is also a very popular technique [3]. In respect of databases, it is Khanna and Zane [4] who first came up with the idea of applying a seal, a digital watermark, while the revolution in the field was brought about by Agrawal and Kiernan [5] in 2002, when the first robust watermarking scheme for relational databases was proposed. Since then, many remarkable techniques have been put forward. However, despite the marked increase in relational databases watermarking schemes, no research has been carried out to date with regard to developing benchmarking tools and platforms that could specialize in evaluating the existing methods.

Benchmarking is the well-known process of a fair performance comparison on a variety of metrics of alternative techniques under the same conditions. The importance of being aware of the comparative strengths and weaknesses of any technique cannot be played down as it allows decisions and choices to be made regarding the suitability of a specific application scenario. The requirement for a benchmark platform for watermarking methods is that it must be able to embed a watermark in a digital asset, to perform various attacks, to try and detect the watermark in a watermarked database table and, finally, all of the above, by running some evaluation metrics towards obtaining a reliable summary of the scheme's performance. While there are numerous benchmark tools in the field of image watermarking, there appears to be a significant lack of benchmark tools for other digital assets.

The current paper proposes the dbMark, which, to the best of the authors' knowledge, is the first benchmarking system of its kind for automatic evaluation and fair comparison of rela-

978-1-5090-5476-3/17/$31.00 ©2017 IEEE

tional databases watermarking schemes. It is a benchmarking tool that covers all of the robust, fragile and fingerprinting types of relational databases watermarking schemes. This tool is expected to help researchers devise new watermarking techniques with better features, and improve existing ones, by investigating the strengths and weaknesses of their proposals. At the same time, it is hoped that this tool will enable application developers to make better informed choices and pick the watermarking scheme best suited to their needs.

The dbMark adapts and incorporates the outstanding characteristics all-round of existing benchmarking tools for multimedia (e.g. a friendly user graphical interface, pluggable architecture, portability, multiple attacks execution, blind watermark extraction, etc.) while it is also comprised of a number of novel features (e.g. the option of embedding multiple watermarks in many data assets at the same time, the automatic parallel performance comparison of alternative schemes, the illustration of the performance characteristics using a rich set of predefined data chart templates, etc.). Moreover, the tool offers all the additional features that are related to the specialized nature and usage of relational databases (database connectivity, updatable content, etc.) and, in addition, it is equipped with the implementation of a large set of attacks that can be carried out against a watermarked database, as well as with various evaluation metrics that can describe the performance of a database watermarking method. The user of the tool can also create and incorporate her/his own features and requirements, for example, by adding new watermarking schemes, new attacks or new evaluation metrics as plugin components.

A prototype of the dbMark has been implemented and is freely available *via* the web to facilitate the experimental work of academics, of the research community or of the software industry. As shown in this present work, the prototype has been tested and thus has verified successfully the performance of at least one technique in each database watermarking category (i.e. robust, fragile and fingerprinting) in comparison to the performance of these techniques as reported in the original research work in which they appeared. This evaluation exercise demonstrates the efficiency of the proposed platform. The implementation of these watermarking techniques is also included in the prototype of the dbMark that is available *via* the Web.

The remainder of this paper develops along the following lines: Section 2 briefly surveys some work carried out on database watermarking methods and some of the main characteristics of existing benchmarking tools for watermarking digital assets other than databases since, to the best of the authors' knowledge, no similar research work appears to have been carried out in this specific field. Section 3 presents the new dbMark benchmarking platform for watermarking relational database methods and discusses all the fundamental concepts that have been taken into consideration in its architectural design. Finally, Section 4 summarizes the paper and gives the impetus for further research.

II. RELATED WORK AND MOTIVATION

The best-known and, in chronological terms, first work to have been carried out in the field of relational databases watermarking was Agrawal and Kiernan in 2002 [5]. The authors propose a robust watermarking scheme for numeric attributes and the embedding of a meaningless watermark using a simple bit-level mechanism. The authors demonstrate that the embedded watermark can survive typical attacks which intend to remove it. Two years on, a very innovative robust watermarking technique for categorical data was proposed by Sion [6]. During the same period Sion et al. [7] proposed a robust watermarking scheme that could be applied to any type of data. To round-off the subject of robust watermarking, Solanas et al. in 2006 [8] put their signature to the first watermarking scheme especially for non-numeric attributes.

Fragile watermarking aims to provide data integrity and verification mechanisms and, for that reason, the embedded watermark should be sensitive to even the smallest unauthorized modification. In chronological terms, a database fragile watermarking scheme first appeared in the literature thanks to Li et al. in 2004 [9]. The proposal is of a distortion-free scheme (i.e. it does not modify any database value in order to embed the watermark) for categorical data, such as social security numbers, names, dates of birth, etc. The scheme is able to localize any unauthorized alteration. The first fragile watermarking scheme that could be applied to any kind of underlying data is owed to the work of Li et al. [10]. Finally, a well-known distortion-based variation of fragile watermarking, which supports data integrity verification and localization as well, was proposed by Guo et al. [11]. This method can also be used for copyright protection purposes.

Fingerprinting belongs to a different class of information-hiding techniques, whereby the embedded watermark is distinct in every distributed copy and provides a treacherous tracing mechanism. Li et al.'s work [12] gave the impetus to research in 2003, with their numeric-based watermarking scheme, and later proposed the best-known fingerprinting technique in the literature [13]. Finally, among many other research works, a significant robust watermarking method that could also be used for fingerprinting was proposed by [14].

A rich bibliography in the field of watermarking schemes for relational databases can be found in the survey [15]. However, despite the fact that a great variety of watermarking techniques are to be found in the literature, a benchmarking tool for the evaluation and fair comparison of databases watermarking schemes had yet to be developed: this has been the aim of the present work. On the other hand, many benchmarking tools for image watermarking schemes exist, the best known being the Strimark [16], which introduces random bilinear geometric distortions to de-synchronize watermarking algorithms. However, the Stirmark presents a number of drawbacks, such as the fact that it does not take into account a watermarking method's false alarm probability (i.e. the probability of detecting a watermark in a non-watermarked image), and the fact that the embedding and detection time are not evaluated. These drawbacks motivated Solachidis et al. to pro-

pose the Optimark [17], which was the first benchmarking tool that included a graphical user interface. Other benchmarks for image watermarking methods are the Checkmark [18] that was initiated in order to provide better evaluation metrics as compared to those of the Stirmark, the OpenWatermark [19], which is a web-based distributed system, the Watermark Evaluation Testbed (WET) [20], which is also web-based, and the OR-Benchmark [21], which incorporates the most recent developments in the field at the time of writing.

By contrast, there is a significant lack of benchmarking tools for other digital assets such as audio, video, software, etc. The only benchmarking tools in existence for audio watermarking methods is an audio version of the Stirmark [22], which can be used for both robust and fragile audio watermarking algorithms, and an audio version [23] of the WET platform. The VidMark [24] is a video watermarking benchmark framework specialized in temporal de-synchronization while [25] is a less specialized benchmark framework for robust video watermarking. Finally, the Mesh Benchmark [26] is a system for the evaluation of robust 3D mesh watermarking techniques.

Although the contribution of benchmarking tools in the field of watermarking is valuable, no research has been carried out as yet towards developing a benchmarking suite for relational databases watermarking schemes. Besides, the existing benchmarking tools for multimedia watermarking are not suitable for application to relational databases, as a result of the utterly different nature of the databases which is comprehensibly explained in [5] (updatable content, less space for the watermark to hide, etc.). This variety in the nature of databases has produced the rich literature on watermarking techniques and attacks that are specialized for relational databases. This perceived need for benchmarking tools was what motivated this present research work the design and implementation of the *dbMark*.

III. THE DBMARK

A. System's Operation and Architecture

The dbMark is a platform-independent benchmarking system dedicated to the evaluation of relational databases watermarking schemes. The system is implemented using Java technology and is therefore easily operational in any operating system that supports Java. The dbMark supports connectivity to MySQL databases using java database connectivity (JDBC) and recognizes database tables with numeric (integers and floating points) and non-numeric (varchar) values. The system's operation follows the well-known 3-phase operation of watermark benchmark platforms to embed a watermark in a digital asset; to perform attacks on the asset; and finally to run a watermark recovery process. Its architecture is illustrated in Figure 1 and it is composed of six components as described below:

Connect: When the dbMark is initialized, the user is prompted to provide all necessary information to connect to a MySQL database (host DNS name, port, username, password and database name). If the database that needs to be tested for watermarking purposes contains sensitive data, such as might pertain to enterprise dealings, funds and customer information, and which needs to be protected, the dbMark is able to connect using the SSL (Secure Sockets Layer) connectivity protocol and exchange the information encrypted. When the connection is established, the user can select a choice of operations in order to proceed: *Watermark*, *Attack*, *Benchmark*, or *Create Plugin*.

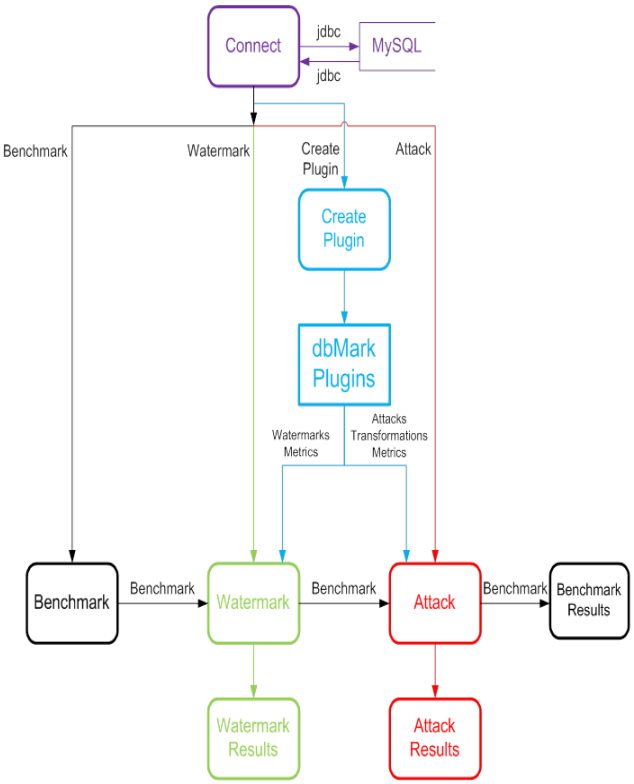

Fig. 1. The architecture of the dbMark.

Watermark: this component prompts the user to choose a watermarking scheme from the existing implementations (plugins), the configuration values for the watermarking scheme's parameters, and the database table that will be watermarked. The benchmarking suite supports single and multiple watermarking, i.e. the watermarking of one table or simultaneously more than one table and the multiple sequential watermarking of a data table using a user-defined sequence of specific watermarking techniques.

Attack: this component enables the user to select and launch an attack on a data table (which can be either watermarked or non-watermarked). Firstly, the user can choose an attack from the dbMark's existing implementations and the targeted data table. Then the user can configure the attack's predefined parameters and the parts of the table that will be attacked, i.e. which fields and/or tuples will be targeted by the attack. The system supports single and multiple attacks, i.e. attacks directed against one or more tables simultaneously and the multiple sequential attack of a data table using a user-defined sequence of specific attacks.

Benchmark: this component is the heart of the dbMark's engine. It utilizes the Watermark and Attack components to measure the efficiency of relational databases watermarking techniques on specific user-predefined datasets. During and after the completion of the operation of the Watermark component, depending on the type of the underlying data (i.e. numeric, alphanumeric or categorical) and also depending on the chosen watermarking scheme, the Benchmark component runs specific metrics to evaluate the scheme performance on the un-attacked underlying relational table, e.g. embedding time; detection reliability; etc. During the Attack operation, the Benchmark component runs metrics that suit the chosen attack to measure its effectiveness, e.g. the attack execution time; the data usability after the attack; etc. Also on the completion of the Attack operation, the Benchmark component runs the watermark recovery process of the chosen watermarking method in order to measure the watermarking scheme's efficiency following the attack, e.g. the execution time for the detection process and its success rate.

Visualization: the dbMark stores the measured benchmark results of all the phases of the Benchmark operation and can illustrate them using the Visualization component into comparative data tables or/and a rich set of flexible data charts, which can be extracted in a text form or/and at an image format, correspondingly. This process is especially useful to researchers who aim to illustrate the evaluation results of multiple schemes or attacks under the same fair environment for comparison purposes.

Create Plugin: to the best of the authors' knowledge, this is an innovative component in digital watermarking benchmark technology. It allows users to create their own plugins and load them into the dbMark's local installation for their personal use. There are three categories of plugins that can be created through this component: Watermark, Attack and Metric plugins. In the creation of a new plugin, the basic information that needs to be specified is the name of the plugin, the type of the underlying data (numeric, alphanumeric and categorical), on which it can be applied, as well as the set of parameters along with their type. Depending on the category of the plugin, the dbMark provides a data form with all kinds of information that also needs to be specified. The input form for a new watermarking plugin will ask of what type the watermarking scheme is (robust, fragile, or fingerprinting) and what the set of its operating parameters is; the input form for a new attack plugin will require the information regarding the set of its operating parameters; and, finally, the input form for building a new metric plugin will ask in which phase the metric will be activated to run, i.e. during the execution of the Watermark, or of the Attack or of the Benchmark components.

B. Available Features

B.1 Watermarking methods

The dbMark prototype comes with a set of implemented watermarking schemes that can be directly used by scholars for training purposes, i.e. to test how to watermark a relational database table, how to evaluate the performance of these watermarking schemes on a given database table, or how to build and load new watermarking schemes into the dbMark engine. The set contains the implementation of one method per watermarking scheme category: robust, fragile, and fingerprinting. Regarding robust watermarking, the dbMark provides an implementation of the watermarking scheme of Agrawal and Kiernan [5], which is the best known robust scheme in the research literature. With regard to fragile watermarking, the most famous scheme [9] is also implemented and ready for use. Finally, with regard to fingerprinting watermarking, the dbMark includes an implementation of the distortion-based method [14] for numeric values, which has been designed as a robust scheme that can also be used as a fingerprinting alternative.

Watermarking schemes can be implemented by users and loaded as plugins onto the dbMark corresponding cartridge. This pluggable architecture is also used for the implementation of the currently available schemes. The users are able to create their own watermarking schemes, with specific parameters and value types, and incorporate them as plugins into the system by using the Create Plugin component. In this way, users may watermark their databases with their own watermarking schemes, which would enable them to benchmark and carry out an evaluation of theirs schemes' performances, in order to determine strengths and weaknesses, and compare them against other available watermarking techniques in the dbMark schemes cartridge.

In order to create a new watermarking scheme plugin, the users have to select the Create Plugin component and to determine the following essential requisites:

- The name of the watermarking scheme’s plugin.
- The scheme’s type: robust, fragile or fingerprinting.
- The underlying data type which the technique can watermark (e.g. numeric or non-numeric values, etc.).

  Consequently, depending on the selected data type, the system will automatically display only the available database tables that are compatible with this scheme.
- The name and data type of the parameters that are needed for the execution of the watermarking method (for example in the case of the watermarking method [5] the users, among other parameters, must specify the number ξ of least significant bits of the data attribute(s) which will be watermarked).

The dbMark will then open a pop-up window with an extractable template within which the users are asked to provide the programming code of a new watermarking scheme, in Java programming language. The code needs to include at least the Encode and Decode functions of the watermarking scheme. The users can import any additional library of programming code and compose any custom method that will be executed by the Encode and Decode functions. In both of these functions the users can return any result (computed during these phases) in a list form, and let the dbMark display it. The users can compile the source code and, in the occurrence of no errors, put the executable file in the corresponding plugin folder so as to be used by the dbMark.

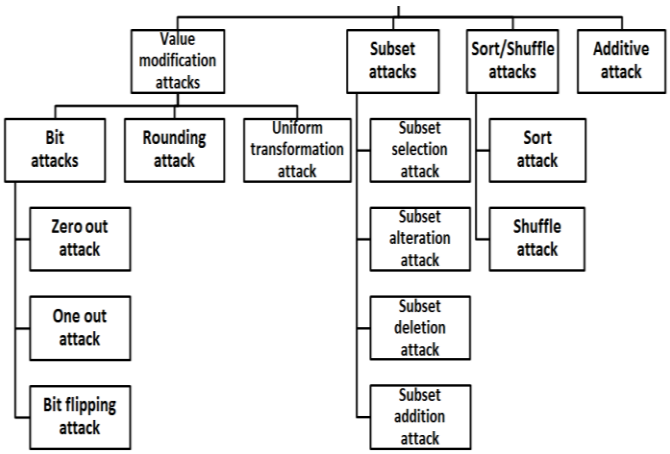

Fig. 2. The set of attacks that are offered by the dbMark.

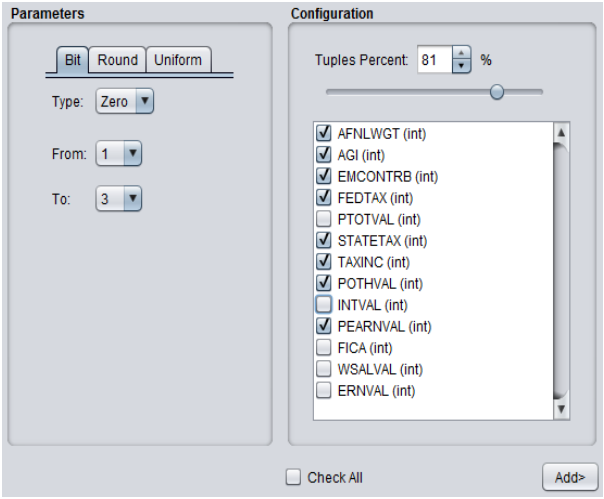

Fig. 3. The preparation for launching a bit attack.

B.2 Attacks

The dbMark prototype comes with a rich set of implemented attacks which can be performed against a watermarked database table. The set of supported attacks is illustrated in Figure 2. The users can configure an attack by choosing its type, a set of necessary parameters and, finally, by specifying the part of the database table that will be attacked, i.e. which attributes and/or tuples. The supported five types of attacks are the following:

a) The *value modification attack* is a type of attack that can be carried out on numeric attributes and is divided into the following categories: i) the *bit attack*, which aims to destroy the watermark by altering some bits of the selected attributes, ii) the *rounding attack*, which aims to remove the watermark by rounding numeric values, and iii) the *uniform transformation attack*, which aims to transform the unit of measurement of the data (e.g. from Fahrenheit to Celsius or Kelvin degrees of temperature) in order to prevent the detection of the watermark. Figure 3 shows the preparation setup phase for launching the *Zero out* bit attack against a selected set of columns of a relational table, by setting to 0 the values of the three right-most least significant bits of 81% of the tuples.

b) The *subset attack* is a type of attack that can be performed on all types of data. It can take one of the following forms: i) the subset selection and the subset deletion attacks, which both aim to prevent the detection of the watermark by means of the selection or of the deletion, respectively, of one subset of the original database table, ii) the subset addition attack, which inserts into the original data table tuples aiming at the desynchronization of the watermark detection process, and iii) the subset alteration attack, which is a combination attack of the subset selection and the value modification attacks.

c) The *sort*/*shuffle attack* is a type of attack that can be carried out against any type of data. The sort attack aims to desynchronize the watermark detection process by interchanging the position of the tuples in a database table on the basis of a specific ordering criterion defined by the user. The shuffle attack swaps the position of tuples in an attempt to destroy the watermark.

d) The *additive attack* is an attack that can be carried out on any type of data. It is a typical false-claim-of-ownership attack in which the malicious user aims to claim ownership of a watermarked relation by embedding her/his own watermark in it. One way of proving ownership is to determine which watermark overlaps with the other (if such an overlap exists) by extracting them both. The "overlapped" watermark (if such a watermark exists) is the one which reveals the real owner of the relation. The dbMark user can carry out the additional attack by selecting the watermarking scheme that will be used to embed a watermark in an already watermarked table.

Besides the choices of implementation available, the users can create their own attacks in the form of a dbMark plugin through the Create Plugin component. The procedure is similar to the corresponding Watermarking schemes plugin: the user needs to select the option to create the type of the attack; to define the necessary parameters for the attack; to extract the resulting template; to compose the attack source code in Java; and to build and provide the executable plugin to the dbMark.

B.3 Metrics for evaluating watermarking algorithms

In order to evaluate a watermarking scheme, the authors have used a variety of evaluation metrics which tend to highlight the strengths and weaknesses comparisons of a scheme against similar techniques. These metrics share a number of common basic characteristics which can be used to group them into categories, such as the underlying types of data they can work with (i.e. numeric, alphanumeric or categorical), the process they are designed to run (i.e. watermark and/or attack) and whether they run during the execution of the process or record the effect of the execution of the process on the data by measuring and comparing the state and condition of the data before and after this execution. The most important evaluation

TABLE I. THE SET OF METRICS THAT ARE OFFERED BY THE DBMARK.

Metric	Form	Underlying data	Process	Phase
Mean change	Plugin	Numeric	Watermark, Attack	Embedding
Mean squared error	Plugin	Numeric	Watermark, Attack	Embedding
Variance change	Plugin	Numeric	Watermark, Attack	Embedding
Data similarity	Plugin	Alphanumeric	Watermark, Attack	Embedding
Watermark capacity	Internal implementation	Implementation	Watermark	Embedding
Detection rate	Internal implementation	Implementation	Watermark	Detection
Execution time	Plugin	All	Watermark, Attack	Embedding, Detection

metrics in the literature include data usability metrics; watermark detection reliability metrics; robustness against typical attacks metrics; as well as the watermark capacity metric and the watermark embedding and detection processes execution time. The dbMark comes with a set of implemented metrics for all of the aforementioned classes that can be used to evaluate and compare watermarking schemes performances. These metrics are summarized in Table I.

The users can also create their own evaluation metrics in two ways: either by using the Create Plugin component to form metric plugins with specific characteristics (for example, by defining the underlying data with which they can work and the process by which they are suitable to run) or by incorporating the evaluation metrics programming code in the code of a watermarking method or of an attack implementation, and to return results that will be recognized and used as black-box values by the benchmarking system. Metric plugins are automatically loaded during a run of any implementation when certain conditions are met (for example, when a specific data type or process is selected by the user for an experimental study), while metrics built to run inside a watermarking scheme or attack will be loaded only during the execution of the corresponding implementation. In the former case the users are able to compare the state and condition of the database before and after the execution, while in the latter they can perform any measurement throughout their process and return the result to the system. The supported evaluation metrics by the dbMark suite follow:

a) Data usability metrics

Preserving the quality of the data is a critically important factor in watermarking and, as a result, many usability metrics have been devised to measure the amount of distortion raised by a watermarking scheme. Most of these metrics measure the distortion by comparing the data before and after watermarking it. The implemented usability metrics of the dbMark include: the mean change, the mean squared error and the variance change metrics, which can all work with numeric attributes (i.e. the metric will be loaded only if the database attribute that is to be watermarked is numeric) as well as the Levenshtein string similarity or distance metric [27], which can work with alphanumeric attributes, measuring the difference between string sequences. All of these metrics have been created in the form of plugins and they are parameterized to run during the watermark embedding and/or attack phases.

b) Watermark capacity metric

The high embedding capacity is an important feature for any watermarking scheme. The watermark capacity metric (also called payload) represents a useful indicator of the amount of information that can be hidden within a database that is intended to be watermarked with a selected watermarking scheme. All watermarking schemes provided by the dbMark measure the watermark capacity during the embedding phase as the percentage of data bits that can be marked without exceeding a predefined limit of acceptable data loss. The users can measure the watermark capacity of their schemes inside the embedding phase of their implementations and return the result to the dbMark.

c) Watermark detection reliability metrics

Every robust and fingerprinting watermarking scheme should indicate a high detection reliability rate by detecting the embedded watermark after an attack. On the other hand, it is of critical importance that a fragile watermarking scheme should be able to detect the entire watermark when no attack takes place and that it should indicate a lower detection reliability rate in any other case. The dbMark measures the detection rate of any watermarking scheme during the corresponding detection phase as the number of the embedded marks detected.

d) Robustness or fragileness against attacks metrics

Robustness, or fragility, against various forms of malicious attacks is the highest priority of any robust and fingerprinting, or fragile, watermarking scheme. In order to measure the resilience or sensitivity of a watermarking scheme, a series of attacks can be carried out covering different attack levels and subsequently detection reliability metrics can be used to calculate the percentage of the original watermark that remained and resisted the attacks. The detection phase of a watermarking scheme in the dbMark system is called by default after any attack execution so the detection rate (i.e. the robustness/fragility against the attack) is measured by all watermarking schemes provided by the dbMark, as well as by any user-implemented watermarking scheme that will be added to the dbMark cartridge.

e) Watermark embedding and detection computational cost

Real world applications require efficient and fast watermarking procedures since nowadays, and in most cases, databases store unprecedented quantities of data. In order to assess the computational cost of a watermarking scheme, the literature suggests calculating the elapsed time or the average number of tuples processed per time unit when the embedding or detection phases are executed. The dbMark includes a metric

plugin that measures the execution time of the watermarking embedding or detection phases.

B.4 Visualization of evaluation results

During any operation, the dbMark collects all evaluation metric results and in the final stage it presents them in tables and flexible charts, such as the ones depicted in Figure 4, all of which can be exported and saved locally. For example, during the Watermark operation, all the Embed/Detect evaluation metric results are collected and, at the end of the process, a panel is displayed with two tabs, Embed and Detect, and each tab contains a set of charts with every one of these charts illustrating the collected values of an evaluation metric. In case a watermarking embedding/detecting process or an attack is executed multiple times, with a different set of parameters values every time, the user can view every evaluation metric results against any parameter selected by drop down menus. In this case, the X axis in a chart represents the different values of the selected parameter and the Y axis represents the evaluation metric results of the execution of the watermarking process or of the attack. If more than one watermarking schemes are evaluated and compared, apart from the schemes' individual panels, an additional comparison panel, of the same structure, is presented at the bottom. This panel's charts enable the user to visualize and compare the results against the parameters that these schemes have in common.

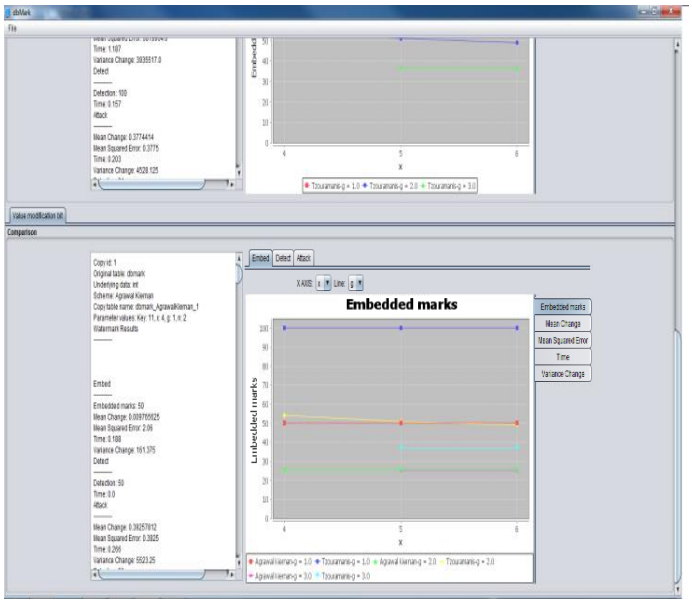

Fig. 4. Results example.

B.5 Datasets

In order to evaluate a watermarking scheme, researchers have used a wide variety of datasets with the appropriate underlying data types, depending on their scheme. The most commonly used dataset for experimentation is the Forest Cover Type dataset, which is a real-life dataset for forest measurements. The dataset consists of a 54-attributes table with 581,012 tuples and it is ideal for watermarking schemes for numeric data. Copyright restrictions prevent the dbMark from redistributing it. However, since license to re-use the dataset is unlimited with retention of original owner and donor copyright notice, interested parties can download the dataset from its web portal [28] and use it.

Another popular real-life dataset is the Census-Income (KDD) dataset, which contains demographic and employment-related data extracted from population surveys conducted by the U.S. Census Bureau. The dataset consists of a 42-attributes table with 299,285 tuples, with a wide variety of underlying data types. Again, due to copyright restrictions, researchers are advised to download the dataset from its portal [29]. Other commonly used datasets that are also available under the same portal are the Wisconsin Diagnostic Breast Cancer dataset, the Iris Plant Dataset and the Abalone dataset.

The dbMark website provides links for downloading and importing all of the aforementioned datasets as database tables into MySQL for the purpose of experimentation with the dbMark.

IV. TEST-RUNS

Some examples of test runs of the dbMark benchmark tool are provided below to show its applicability and efficiency in evaluating and comparing the performance of watermarking methods for relational data. With regard to robust watermarking, the study tests the implementation provided in the cartridge of the dbMark for the Agrawal and Kiernan scheme [5]. Similar tests are presented for the implementation provided in the cartridge of the dbMark for the Tzouramanis scheme [14], which can be used either as a robust or as a fingerprinting method. A 1,024 bits message is chosen to take on the role of the meaningful watermark for this watermarking scheme. The experimental study also examines the performance of the implementation provided by the dbMark for the fragile watermarking scheme introduced by Li et al. in [9] for numeric and alphanumeric categorical data. An average number of 100 tuples is set to belong in every group for which this watermarking scheme divides the data securely before watermarking them.

The relational tables which were used for the experimentation and which were to be watermarked in all the experiments are the Forest Cover Type and the Census-Income datasets mentioned earlier. The settings that are summarized in Table II were used. Every experiment is repeated ten times and the average values of the measured parameters are calculated. The results are also verified through a comparison with the results presented in the original papers introducing the three above database watermarking schemes. The workstation in which the experiments are taking place is equipped with Intel Core i5 CPU running at 2.7GHz with a 16Gbytes RAM and Windows 7 Pro 64-bit.

The first set of experiments evaluates the efficiency of the watermark encoding process for all three watermarking schemes. The dbMark platform is set to watermark every single value in the Elevation attribute of the Forest Cover Type

TABLE II. PARAMETER VALUES USED IN THE EXPERIMENTS.

Dataset:	**Forest Cover Type**	**Census-Income**
Dataset size (in tuples):	581,012	299,285
Tuples to be marked (in %):	100	
Column to be marked:	Elevation (numeric)	Age (numeric / categorical)
Primary key:	an extra column is added to serve as the primary key	
Least significant bits (in number):	6	not applicable

dataset. The number of least significant bits available for marking for the Agrawal and Kiernan and for the Tzouramanis schemes is set to be equal to 6. The results in Figure 5 show that the watermarking embedding process does not affect significantly the mean and variance on the data (the Li et al. fragile scheme is distortion-free). The results for the Agrawal and Kiernan scheme also fall in harmony with the findings in the original work of [5].

Watermarking method:	**Agrawal and Kiernan**	**Tzouramanis**	**Li et al.**
Execution time (sec):	479.7	6,588	13,332
Mean change (%):	0	0	–
Variance change (%):	0.019	0.072	–

Fig. 5. A comparison of the efficiency of the watermarking embedding phase of three database watermarking methods, using the dbMark tool.

As regards the time factor for embedding the watermark, the scheme by Agrawal and Kiernan points to much a lower cost than the other two schemes since, for every tuple, it involves the computation of a single cryptographically secure one-way hash function, while the Tzouramanis scheme involves more than one hash function computation and, finally, the Li et al. scheme involves much more demanding computation. In addition, the last two schemes divide the tuples into groups according to some security parameters, and then mark every group independently, while the Agrawal and Kiernan scheme marks every single tuple separately using a meaningless watermark. In comparison to the Tzouramanis scheme, the impossibility of embedding a meaningful watermark is the only major drawback in this set of experiments for the Agrawal and Kiernan scheme.

The second set of experiments assesses the efficiency of the watermarking detection process when no attack has taken place for the Forest Cover Type dataset. Figure 6 illustrates the results. With regard to the time needed to perform this operation, it is worth noting that the major cost for the Agrawal and Kiernan method is the computation of a single hash function which is needed to determine the presence of the mark in every tuple. On the other hand, the Tzouramanis and the Li et al. schemes require much more complicated checks during the watermark detection process.

Watermarking method:	**Agrawal and Kiernan**	**Tzouramanis**	**Li et al.**
Execution time (sec):	1.773	4.945	138.60
Marks detected (%):	100.0	100.0	99.87

Fig. 6. A comparison of the efficiency of the watermarking detection phase of three database watermarking methods when no attack has taken place, using the dbMark tool.

The experiments that follow study the effectiveness and practicability of the dbMark platform in executing attacks against watermarked relational databases and in measuring their effect on the protected data.

The first set of this class of experiments using the Forest Cover Type dataset evaluates the efficiency of the implemented robust watermarking scheme of Agrawal and Kiernan when exposed to a number of attacks. By assuming that the watermark has been spread across the six right-most (i.e. least significant) bits of the data, three types of attacks are considered: i) a value modification attack in which the value of the right-most (or of the two or three right-most, respectively) of the least significant bits of the marked data is randomly flipped, ii) a subset attack in which 50% of the tuples are randomly selected and deleted, and, iii) an additive attack in which a malicious watermark is encoded into a previously legitimately watermarked relation using the same input parameters, i.e. in the six right-most least significant bits of the data.

Attack:	**Value modification attack**			**Subset attack**	**Additive attack**
Bits flipped (in number):	1	2	3	0	0
Tuples deleted (%):	0	0	0	50	0
Execution time (sec):	5,312	4,998	4,937	2,180	485.6
Detection success rate (%):	100	67	66	100	91

Fig. 7. The efficiency of the Agrawal and Kiernan robust watermarking method under several types of attacks.

For every case, the time needed to execute the attack, and the watermark detection success rate, are measured. The results of the experiments in Figure 7 can be verified through the findings of the original work of [5] whereby the watermark can be detected even if the level of distortion on the data, or the percentage of the deleted tuples, are high.

For a comparison with the performance of the Agrawal and Kiernan robust watermarking scheme, Figure 8 illustrates the corresponding performance of the Tzouramanis scheme under the same attacks and parameter settings. The time necessary to execute the value modification and subset deletion attacks in both Figures 7 and 8 are comparable, since it depends mainly on the underlying dataset and the parameter settings, which are the same. However the time necessary to execute the additive attack in Figure 8 is much higher since it is assumed that the attacker encodes its watermark using the same watermarking scheme as the legitimate owner of the data. The watermark detection success rate of the Tzouramanis scheme, subsequent-

Attack:	Value modification attack			Subset attack	Additive attack
Bits flipped (in number):	1	2	3	0	0
Tuples deleted (%):	0	0	0	50	0
Execution time (sec):	4,793	5,467	5,887	2,527	6,162
Detection success rate (%):	100	93	63	94	92

Fig. 8. The efficiency of the Tzouramanis robust watermarking method under several types of attacks.

ly to the value modification and subset attacks, are similar to the ones reported in the original work of [14], which proposes the scheme (for example, in Figure 10(a) of [14] it is shown that a subset deletion attack of 50% of the tuples of a watermarked relation with a watermark size equal to 1Kbits gives a watermark detection success rate that is close to 100%).

The last set of experiments examines the efficiency of the fragile watermarking scheme of Li et al. in detecting two types of attacks, the subset deletion and the shuffle attack. The underlying data is a watermarked version of the categorical "Age" attribute of the Census-Income relation. Every attack is repeated ten times and, as Figure 9 indicates, the fragile watermarking scheme detects all of them, even if the damage caused to the watermark is not significant. The results obtained about the efficiency of the watermarking scheme can be verified through the analytical results that appear in the original work that introduces the watermarking scheme (for example, on the right of Figure 7 of [9] it is shown that the subset deletion attack of 10% of the tuples of a watermarked relation is expected to go undetected by the fragile watermarking scheme, with a probability of under 10^{-5}).

Attack:	Subset attack	Shuffle attack
Tuples deleted (%):	10	0
Tuples shuffled (%):	0	10
Attack execution time (sec):	89.37	210.19
Attacks detection success rate (%):	100	100

Fig. 9. The efficiency of the Li et al. fragile watermarking method under several types of attacks.

The most important conclusion of the above test-runs for this work is that the dbMark offers (i) effective, fast and easily manageable watermarking embedding and detection modules and (ii) a rich set of metrics to measure the robustness or fragility of any implemented relational database watermarking method against the attacks that can be launched against the protected data. This points to the dbMark as an efficient benchmarking software tool that allows its use for the experimental study and comparison of the efficiency of relational data watermarking methods.

V. CONCLUSION

This paper investigates the issue of benchmarking watermarking methods for relational data. It presents the first attempt specifically designed for the automatic evaluation and comparison of watermarking methods from this field of digital data. The new tool is called the dbMark, and its basic concepts and operations are examined here.

The dbMark is very parametric and flexible in benchmarking all kinds of watermarking methods (i.e. robust, fingerprinting and fragile) for relational databases, through the use of a set of implemented attacks and by measuring a series of relevant evaluation metrics. The user of the tool can also create and incorporate her/his own features and requirements, for example, by adding new watermarking methods, attacks or evaluation metrics as plugin components. The experiments that have been presented indicate that the new tool can be used effectively as a primary platform for the experimental evaluation and fair comparison of watermarking methods for numeric, alphanumeric and categorical relational data.

The open-source prototype implementation of the dbMark is freely accessible on the Web[1]. Future plans are to enrich the tool's platform with the implementation of more available features, i.e. watermarking methods, attacks and evaluation metrics. In addition, the world of the social web provides the inspiration for the future planning of the dbMark central server which would allow users to submit their own plugins to the system through their personal account, with the aim of sharing these with other users. In this way, the dbMark user community would be able to put every proposed prototype implementation to the test and to eventually provide its creator with feedback that would contribute to its improvement. A single central exchange point for the watermarking database community such as this one could contribute to the dbMark becoming a non-negligible tool to be consulted by researchers, enabling them at the same time to promote their own proposals by putting these forward through the dbMark suite for evaluation and comparison by the remainder of the research community.

REFERENCES

[1] H. Hacigumus, I. Bala, and M. Sharad, "Providing database as a service", In *Proceedings of the 18th IEEE International Conference on Data Engineering (ICDE)*, pp. 29-38, 2002.

[2] F.M. Boland, J.J. O'Ruanaidh, and C. Dautzenberg, "Watermarking digital images for copyright protection". In *Proceedings of the 5th International Conference on Image Processing and its Applications*, pp.326-330 (1995).

[3] K. Nahrstedt, L. Qiao, and I. Dittmann, "Non-invertible watermarking methods for MPEG video and audio". In *Proceedings of the Security Workshop at ACM Multimedia*, pp.93-98 (1998).

[4] S. Khanna, and F. Zane, "Watermarking Maps: Hiding Information in Structured Data". In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp.596-605, (2000).

[1] The dbMark benchmarking suite and its plugins are freely available for sharing and are provided via the web at the address https://www.icsd.aegean.gr/t.tzouramanis/papers/dbMark/.

[5] R. Agrawal, and J. Kiernan, "Watermarking Relational Databases". In *Proceedings of the 28th International Conference on Very Large Databases (VLDB)*, pp.155-166 (2002).

[6] R. Sion, "Proving Ownership over Categorical Data". In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pp.584-596 (2004).

[7] R. Sion, M.J. Atallah, and S. Prabhakar, "Rights Protection for Relational Data". *IEEE Transaction on Knowledge & Data Engineering*. 16(12), pp.1509-1525 (2004).

[8] A. Solanas, and J. Domingo-Ferrer, "Watermarking Non-numerical Databases". In *Proceedings of the 3rd International Conference on Modeling Decisions for Artificial Intelligence (MDAI)*, pp.239-250 (2006).

[9] Y. Li, H. Guo, and S. Jajodia, "Tamper detection and localization for categorical data using fragile watermarks". In *Proceedings of the Digital Rights Management Workshop*, pp.73-82 (2004).

[10] Y. Li, and R. Huijie Deng, "Publicly verifiable ownership protection for relational databases". In *Proceedings of the Proceedings of the ACM Symposium on Information, Computer & Communications Security (ASIACCS)*, pp.78-89 (2006).

[11] H. Guo, Y. Li, A. Liu, and S. Jajodia, "A fragile watermarking scheme for detecting malicious modifications of database relations". *Information Sciences*, 176(10), pp.1350-1378 (2006).

[12] Y. Li, V. Swarup, and S. Jajodia,: "Constructing a virtual primary key for fingerprinting relational data". In *Proceedings of the Digital Rights Management Workshop*, pp.133-141 (2003).

[13] Y. Li, V. Swarup, and S. Jajodia, "Fingerprinting Relational Databases: Schemes and Specialties". *IEEE Transactions on Dependable & Secure Computing*, 2(1), pp.34-45 (2005).

[14] T. Tzouramanis, "A Robust Watermarking Scheme for Relational Databases", In *Proceedings of the 6th IEEE International Conference on Internet Technology & Secured Transactions (ICITST)*, pp.783-790 (2011).

[15] R. Halder, S. Pal, and A. Cortesi, "Watermarking Techniques for Relational Databases: Survey, Classification and Comparison". *Journal of Universal Computer Science*, 16(21), pp.3164-3190 (2010).

[16] F.A.P. Petitcolas, R.J. Anderson, and M.G. Kuhn, "Attacks on Copyright Marking Systems". In *Proceedings of the 2nd International Workshop on Information Hiding*, pp.218-238 (1998).

[17] V. Solachidis, A. Tefas, N. Nikolaidis, S. Tsekeridou, A. Nikolaidis, and I. Pitas, "A Benchmarking Protocol for Watermarking Methods". In *Proceedings of the International Conference on Image Processing (ICIP)*, Vol.3, 1023-1026 (2001).

[18] S. Pereira, S. Voloshynovskiy, M. Madueno, S. Marchand-Maillet, and T. Pun, "Second Generation Benchmarking and Application Oriented Evaluation". In *Proceedings of the 4th International Workshop on Information Hiding*, pp.340-353 (2001).

[19] B. Michiels, and B. Macq, "Benchmarking image watermarking algorithms with Open Watermark", In *Proceedings of the 14th European Signal Processing Conference (EUSIPCO)*, pp.4-8 (2006).

[20] H. Cook Kim, H. Ogunleye, O. Guitart, and E.J. Delp, "The Watermark Evaluation Testbed (WET)". In *Proceedings of the Security, Steganography & Watermarking of Multimedia Contents Conference (SSWMC)*, pp.236-247, 2004.

[21] H. Wang, A.T. Ho, and S. Li, "OR-Benchmark: An Open and Reconfigurable Digital Watermarking Benchmarking Framework". arXiv preprint arXiv:1506.00243 (2015).

[22] J. Dittmann, M. Steinebach, A. Lang, and S. Zmudzinski, "Advanced Audio Watermarking Benchmarking". In *Proceedings of the Security, Steganography & Watermarking of Multimedia Contents Conference (SSWMC)*, pp.224-235 (2004).

[23] A. Lang, J. Dittmann, E.T. Lin, and E.J. Delp, "Application-oriented audio watermark benchmark service". In *Proceedings of the Security, Steganography & Watermarking of Multimedia Contents Conference (SSWMC)*, pp.275-286 (2005).

[24] P.A. Hernandez-Avalos, C. Feregrino-Uribe, R. Cumplido, and J.J. Garcia-Hernandez, "Towards the Construction of a Benchmark for Video Watermarking Systems: Temporal Desynchronization Attacks". In *Proceedings of the 53rd IEEE International Midwest Symposium on Circuits & Systems*, pp. 628-631 (2010).

[25] P. Schaber, S. Kopf, C. Wesch, and W. Effelsberg, "CamMark: a camcorder copy simulation as watermarking benchmark for digital video". In *Proceedings of the 5th ACM Multimedia Systems Conference (MMSys)*, pp.91-102, (2014).

[26] K. Wang, G. Lavoue, F. Denis, A. Baskurt, and X. He, "A Benchmark for 3D Mesh Watermarking". In *Proceedings of the Shape Modeling International Conference (SMI)*, pp.231-235 (2010).

[27] V.I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals". *Soviet Physics Doklady*, 10(8), pp. 707-710 (1966).

[28] University of California Irvine KDD Archive: The Forest Cover Type Dataset, available at: http://kdd.ics.uci.edu/databases/covertype/covertype.html, valid as of April 2017.

[29] University of California: Machine Learning Repository, available at: http://archive.ics.uci.edu/ml/datasets.html, valid as of April 2017.