Watermarking scheme evaluation tool

Nazim Fatès and Fabien A. P. Petitcolas
Microsoft Research
n_fates@scientist.com, fabienpe@microsoft.com

Abstract

Digital watermarking has been presented as a solution for protection against illegal copying of multimedia objects and dozens algorithms have been proposed. Some problems seriously darken the future of this technology though. One of them is that the requirements, tools and methodologies to assess the current technologies are almost inexistent. The lack of benchmarking of current algorithms is blatant. This confuses rights holders as well as software and hardware manufacturers and prevents them from using the solution appropriate to their needs. Indeed basing long-lived protection schemes on badly tested watermarking technology does not make sense.

In this paper we will present the architecture of an evaluation tool being developed.

1. Need for evaluation

Digital watermarking remains a largely untested field and only very few large industrial consortiums have published requirements against which watermarking algorithms should be tested. For instance the International Federation for the Phonographic Industry led one of the first large scale comparative testing of watermarking algorithm for audio [1]. In general, a number of broad claims have been made about the 'robustness' of various digital watermarking or fingerprinting methods but the growing number of attacks against such systems (e.g., [2, 3]) has shown that far more research is actually required to improve the quality of existing watermarking methods.

With a common benchmark authors and watermarking software providers would just need to provide a table of results, which would give a good and reliable summary of the performances of the proposed scheme [4]. So end users can check whether their basic requirements are satisfied, researchers can compare different algorithms and see how a method can be improved or whether a newly added feature actually improves the reliability of the whole method and the industry can properly evaluate risks associated to the use of a particular solution by knowing which level of reliability can be achieved by each contender.

2. Evaluation tool

As a first step towards a widely accepted way to evaluate watermarking schemes we started to implement an automated benchmark server for digital watermarking schemes. The idea is to allow users to send a binary library of their scheme to the server which in turns runs a series of tests on this library and keeps the results in a database accessible to the scheme owner or to all 'watermarkers' through the Web.

Simplicity—In order to be widely accepted this service has a simple interface with existing watermarking libraries (only three functions must be provided). It also takes into account the application of the watermarking scheme by proposing different evaluation profiles (sets of tests and images) and strengths. These goals a reflected in Figure 1 and will be detailed in the next sections.

Customisation—For each type of watermarking scheme, we want to use a different evaluation profile without having to recompile the application tool. Definition of the profiles is not an easy task and requires agreement among the watermarking community. As we will see however, the choice of these profiles does not affect the design of the evaluation service and can be done later and tuned after experimenting the service.

Modularity and choice of tests—Watermarking algorithms are often used in larger system designed to achieve certain goals (e.g., prevention of illegal copying, trading of images). But here we are only concerned with the evaluation of watermarking (so the signal processing aspects) within the larger system not the effectiveness of the full system to achieve its goals. So the main functionalities we wish to evaluate include the perceptibility of the scheme, its capacity, its reliability (robustness to attacks and false alarm rate) and its performances (mainly the speed of execution). For each of these set of tests we have implemented ad-hoc libraries which are built easily on top of the core libraries as shown in the next section.

- Perceptibility characterises the amount of distortion introduced during by the watermarking scheme itself. The problem here is very similar to the evaluation of compression algorithms. We allow the addition and use of different quality metrics, the simplest and most widely used one being the P.S.N.R.
- The capacity of a scheme is the amount of information one can hide. In most applications the capacity will be a fixed constraint of the system so robustness test will

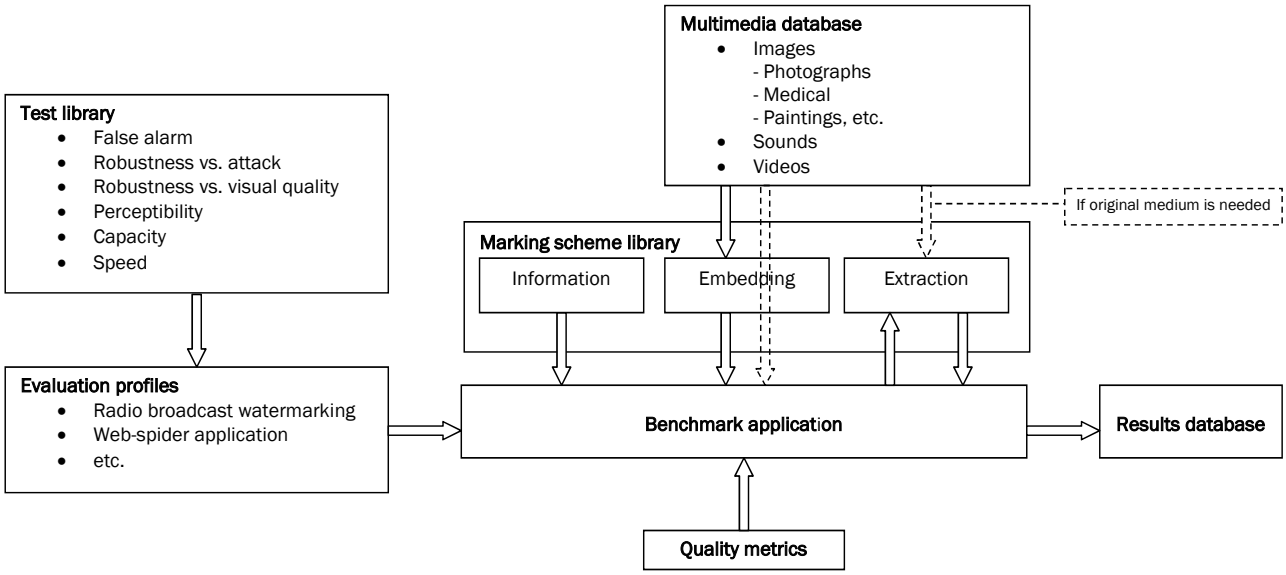

Figure 1—Data flow for the watermarking evaluation service. The marking scheme is provided by the user as a library of functions. This library exports in particular an information function which is used to select which evaluation profile has to be used. The evaluation profile is composed of a list of tests or attacks to be applied and a list of multimedia object required for the test and sorted by types and categories. All test results are uploaded to an SQL server connected to a Web server.

be done with a random payload of given size. However knowing the relation between capacity and robustness is very important and our benchmark provide a test that help to analyse this trade-off by drawing different graphs [4].

- The robustness can be assessed by measuring the detection probability of the mark and the bit error rate for a set of criteria that are relevant to the application, which is considered. Part of these evaluation profiles can be defined using a finite and precise set of robustness criteria (e.g., S.D.M.I., IFPI or E.B.U. requirements) and one just needs to check them.
- False alarms are difficult to measure and we are working on a method to estimate them automatically without having to do an exhaustive search of the key space.
- Finally, speed is very dependent on the type of implementation: software or hardware. Here we are only concerned with software implementation and our test just computes an average of the time required on a particular given platform to watermark and image depending on its size.

3. The architecture

3.1. Simple interface

The evaluation service only requires three functions to be exported from the watermarking library supplied by the user. The first one, *GetSchemeInfo* provides information about the marking scheme such as its type and purpose, its operational environment, its author, version, release date, etc. The two other functions are the complementary *Embed* and *Extract* functions.

We tried to capture all possible cases and ended up with a solution where several parameters are provided but not all of them are mandatory. To make the addition of new parameters easy, these parameters have been encapsulated into a single structure. They include the original medium, the watermarked medium, the embedding key, the strength of the embedding, the payload, the maximum distortion tolerated and the certainty of extraction. This very simple technique allows interoperability with schemes of various types and only requires having a common unique source code header to maintain.

3.2. Profiles

This is achieved with the use of an initialisation file per evaluation profile, in which each test has its own parameters stored. Table 1 gives an example of two different initialisation files that could correspond to the evaluation profile of a blind watermarking scheme that applies to radio broadcasting and to the evaluation profile of a non-blind watermarking scheme that applies to images for proof of ownership.

3.3. Class structure

The project is being written using the C++ language to take full advantage of the inheritance and polymorphism features of an object-oriented language. Indeed, one of the ambitions of the StirMark Benchmark project is to provide a single tool that will be able to test different kinds of media such as images, sounds and videos. A

Table 1—Initialisation file samples.

Blind audio watermarking	Non blind image watermarking
[Test list] Test 1=Mean embedding time Test 2=Mean extraction time Test 3=Sound Low pass filter [Mean embedding time] Number of tests=100 [Mean extraction time] Number of tests=100000 [Sound Low pass filter] Cut frequency=2000 [Samples] Set 1=Radio broadcasts Set 2=Voices Set 3=Songs	[Test list] Test 1=Mean embedding time Test 2=Noise addition Test 3=Image JPEG compression [Mean embedding time] Number of tests=100000 [Noise addition] Noise start level=0.25 Noise end level=0.75 Step=0.05 [Image JPEG compression] Quality start=100 Quality end=75 Step=5 [Database] Set 1=Medical pictures Set 2=Photographs

UML simplified representation of the architecture is provided in .

CBench is the general wrapper class for all possible benchmark. It creates a list of tests and a list of mediums images according to the evaluation profile being used. It is also responsible for the management of the watermarking libraries and creates a *CMarkingScheme* object. This latter class acts as an interface between the evaluation service object model and the libraries provided by the users. *CMedium* is a base class used to handle medium data and in particular memory allocation.

CTest takes a list of media and a marking scheme as an input and performs a test on it. The test can be whatever we need to evaluate the basic functionalities. Typical tests are false-alarm tests, embedding time and robustness tests (embedding, transformation, extraction). At last *CImageTransformation* performs a transformation on a medium (an image as shown on the figure). For images, these include filtering, geometric transformation and any other distortion required for testing.

Although the marking methods vary from one medium to another, many tests are common to all the media. For example a robustness test can be expressed as follows:

- For each medium in a determined set:
  1. Embed a random payload with the greatest strength, which does not introduce annoying effects. In other words, embed the mark such that the quality of the output—for a given quality metric—is greater than a given minima.
  2. Apply a set of given transformations to the marked medium.
- For each distorted medium try to extract the watermark and measure the certainty of extraction.
- The measure for the robustness is the certainty of detection or the bit error rate after extraction.

This procedure, which is parameterised in the evaluation profile, must be repeated several times since the hidden information is random and a test may be successful by chance. It is clear that such a test does not have to be specific to a certain type of medium and can be written using a *CMedium* base class. However the transformations used in the test must be aware of the type of medium. For instance a geometric transformation of an image has to be written using the derived class *CImage* as it needs functions specific to image processing such as *GetPixel* for instance.

The advantage of using an object-oriented structure is clearly to simplify the use of the software. For instance if one wants to add a new attack, say adding noise to a still image, then one needs to create a *CAddNoise* object, derived from the abstract class *CImageTransform*, and this can be done by writing one or two lines of code describ-

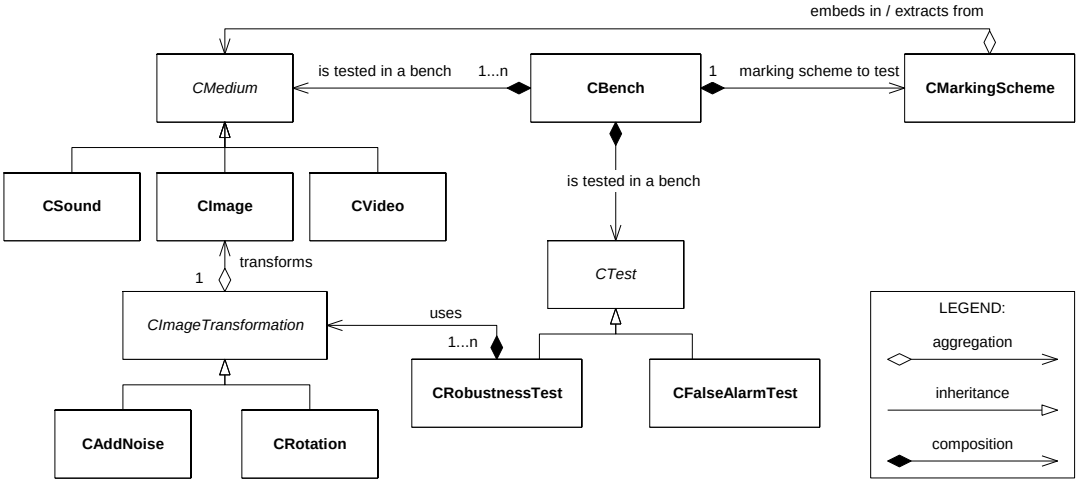

Figure 2—Simplified class diagram of the core of the evaluation tool.

ing how to change the value of the pixels to add the noise. The same mechanisms can be applied to the writing of new tests. The object-oriented structure is used her to handle all the 'administrative tasks' such as reading the media in the database (this is done by the *CBench* class), embedding the mark (this is done by the *CMarking* class), exporting results, etc. This means that the researchers can focus their energy in writing appropriate code for attacks and tests without having to care for the management. The application then can generate a broad range of results and plots like the one shown in Figure 3.

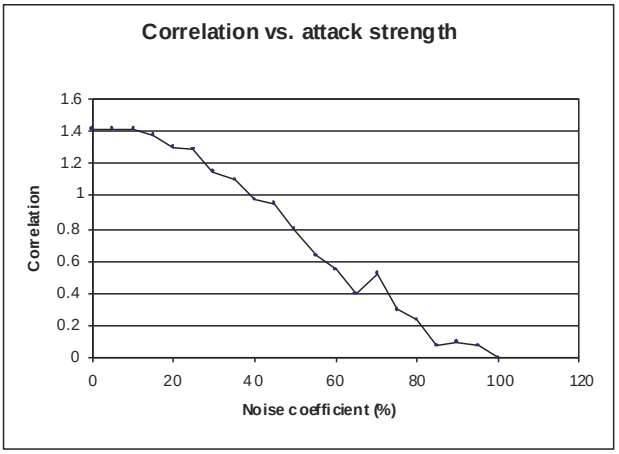

Figure 3—Example of robustness test graph showing the trade-off between robustness (in this correlation computed by the detector) and attack strength (% of noise added to the image). Algorithm: simple spread-spectrum technique. Sample image: Lena.

4. Conclusions

In this paper we have described the architecture of a fully automated evaluation tool for digital watermarking schemes. It is the logical continuation of the early benchmark introduced into StirMark [5]. This new benchmark is operated on a piece of code that is provided by the user through a library and uses an object-oriented language to make multimedia handling quite simple. It also relies on pre-defined evaluation profiles (configuration files), allowing testing of different types of watermarking schemes automatically to different levels of assurance.

Hopefully this new generation of watermarking testing tool will be very useful to the watermarking community as it will provide a standard for testing and allow fair comparison between different watermarking schemes!

5. References

1 International Federation of the Phonographic Industry. Request for Proposals – Embedded Signalling Systems Issue 1.0. 54 Regent Street, London W1R 5PJ, June 1997.

2 Martin Kutter. Watermark copy attack. In Ping Wah Wong and Edward J. Delp, editors, proceedings of Electronic imaging '99, security and watermarking of multimedia contents II, vol. 3971, pp. 371–380, San Jose, California, U.S.A., 24–26 January 2000. The Society for imaging science and technology (IS&T) and the international Society for optical engineering (SPIE).

3 Fabien A. P. Petitcolas, Ross J. Anderson and Markus G. Kuhn. Attacks on copyright marking systems. In David Aucsmith, editor, second workshop on information hiding, in vol. 1525 of Lecture notes in computer science Portland, Oregon, U.S.A., 14–17 April, 1998, pp. 218–238. ISBN 3-540-65386-4.

4 Martin Kutter and Fabien A. P. Petitcolas. A fair benchmark for image watermarking systems. In Ping Wah Wong and Edward J. Delp, editors, proceedings of Electronic imaging '99, security and watermarking of multimedia contents, vol. 3657, pp. 226--239, San Jose, California, U.S.A., 25--27 January 1999. The Society for imaging science and technology (IS&T) and the international Society for optical engineering (SPIE). ISSN 0277-786X. ISBN 0-8194-3128-1.

5 http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark/