IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

D.WVL.4
First Summary Report on Asymmetric Watermarking

Due date of deliverable: 31. January 2005
Actual submission date: 31. January 2005

Start date of project: 1 February 2004 Duration: 4 years

Lead contractors: Centre National de la Recherche Scientifique (CNRS), Otto-von-Guericke Universität Magdeburg (GAUSS)

Revision 1.0

Project co-funded by the European Commission within the 6th Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

First Summary Report on Asymmetric Watermarking

Editors
Patrick Bas (CNRS)
Stefan Katzenbeisser (GAUSS)

Contributors
André Adelsbach (RUB)
Mauro Barni (CNIT)
Patrick Bas (CNRS)
Stefan Katzenbeisser (GAUSS)
Alessia De Rosa (CNIT)
Ahmad-Reza Sadeghi (RUB)

31. January 2005
Revision 1.0

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932.

Contents

1 Introduction **1**

1.1 Why Asymmetric Schemes? 1

1.1.1 Public Key Watermarking 3

1.1.2 Asymmetric Watermarking 4

1.2 Asymmetric Versus Zero-Knowledge Watermarking 4

2 Asymmetric Watermarking **5**

2.1 Asymmetric Watermarking Using Matrix Products 5

2.1.1 Key Independent Watermark Detection 5

2.1.2 Public Key Watermarking by Eigenvectors of Linear Transforms . . . 6

2.2 Asymmetric Watermarking Using Spectrum Constraints 8

2.3 Unified Approach with Quadratic Detection 9

2.4 Linear Asymmetric Watermarking Schemes 10

2.4.1 Partial Key Embedding System 11

2.4.2 Transformed-Key Watermarking System 11

2.4.3 Private Keys Generation Using Phase-shift-transforms 11

2.5 A Critical View of Asymmetric Watermarking: Misconceptions and Potentials 12

2.5.1 Early Algorithms: the Wrong Approach 12

2.5.2 Perspectives for Future Research 14

3 Zero-Knowledge Watermarking **15**

3.1 Zero-Knowledge Watermark Detection Protocols 15

3.1.1 Interactive Proof Systems 16

3.1.2 Zero-Knowledge Property 17

3.1.3 Design of Zero-Knowledge Watermark Detectors 19

3.1.4 Comparison of Zero-Knowledge Watermark Detectors 20
3.1.5 Early Approaches to Zero-Knowledge Watermarking 21
3.2 Computing with Committed Values 22
3.2.1 Building Blocks 23
3.2.2 Protocol 24

Bibliography **27**

Chapter 1

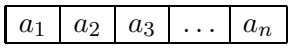
Introduction

1.1 Why Asymmetric Schemes?

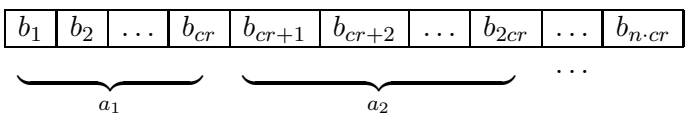
Traditional watermarking schemes—as found in the literature [11]—are essentially *symmetric*, which means that the same key is used both in the watermark embedding and detection process. Similar to symmetric ciphers, this key must be considered critical to the security of the watermarking scheme.[1] Once the key is known to an attacker, watermarks can be removed from digital objects easily. This fact limits the usability of watermarks. In a typical application, a watermark, representing certain application-dependent information, is embedded into a digital object. Later, a party called *prover* proves to a *verifier* that this watermark is indeed detectable in some possibly modified version of the content. In many cases the verifier cannot be fully trusted, which means that sensitive information (especially the watermarking key) should not be disclosed to him.

This problem could be resolved by asymmetric watermarking systems. Similar to public key cryptography, asymmetric schemes allow watermarks to be embedded using a private key. However, the watermark extraction process relies on a different key (called a public key), which contains enough information to successfully prove the presence of a watermark but does not contain enough information to remove the private watermark.

Traditionally, the watermark verification process requires the complete disclosure of the secret watermarking key. Consider, for example, a classic watermarking scheme by Hartung and Girod [26], who developed a technique to watermark digital video based on spread spectrum signals in the spatial domain. Let $a_j \in \{-1, 1\}$ be the watermark, encoded as strings of 1 and -1, to be hidden in a video stream v_i:



A sequence b_j is produced out of a_i by repeating each sequence element cr times:



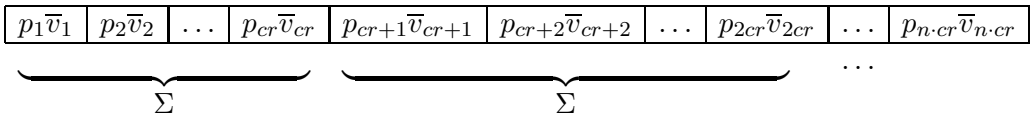
[1] In many schemes, both the watermark and the key will be considered security critical because the private key is often used to generate the string which is embedded as watermark.

Formally, b_j is a sequence of length $n \cdot cr$ such that $b_i = a_j$ for all indices i and j with $j\,cr \leq i < (j+1)\,cr$. The new sequence b_i is multiplied by a pseudo-noise sequence $p_i \in \{-1, 1\}$, scaled by a constant α and added to the video stream to be watermarked:

$$\overline{v}_i = v_i + \alpha b_i p_i.$$

Here, $\overline{v}_i$ denotes the watermarked video stream. Due to the noisy appearance of p_i, the watermark $\alpha b_i p_i$ is also noise-like and therefore difficult to detect and remove.

In order to verify the mark, the sequence p_i used in the embedding process must be known; the possibly modified video stream $\overline{v_i}$ is multiplied by the same sequence p_i that was used in the embedding process. After multiplication, all sequence elements corresponding to one specific watermarking bit are added:



Formally,

$$s_j = \sum_{j\,cr \leq i < (j+1)\,cr} p_i \overline{v_i} \quad \approx \quad \sum_{j\,cr \leq i < (j+1)\,cr} p_i^2 \alpha b_i.$$

Assuming that the pseudo-noise signal p_i and the video stream v_i are uncorrelated, the sum should be close to $s_j \approx \alpha\,cr\,a_j$ and a_j can be recovered by $a_j = \mathrm{sign}(s_j)$. To correctly decode the secret information, only the sequence p_i (which forms the watermarking key) must be known; thus, this system is an example of a blind watermarking scheme. If a different sequence is used, the recovered watermark bits are random.

In the scheme depicted above, the watermarking key consists of the sequence p_i (or a seed to a pseudo-random number generator that produces p_i). In many watermarking systems the watermark key specifies the location of the watermark in the digital data or contains sufficient information to remove the watermark completely. In the watermarking system above, an attacker can simply subtract the sequence $p_i \alpha b_i$ from the watermarked video signal, once he knows both the watermark and the key. This operation completely removes the watermark.

From the perspective of a protocol designer, the watermarking system may thus be considered secure as long as there is no need to verify the watermark; once the mark is disclosed in a protocol, the mark can be removed by the party who gains access to the watermark key. If several digital objects were watermarked with the same mark and key, those other objects are at risk, too.

Another important aspect is the usability of symmetric schemes: Knowledge of the symmetric key is necessary to determine whether a watermark is present in a digital object. However, this prevents the mark from being used for detection by third parties, e.g., if a potential customer wishes to determine the owner of an unlabeled image or piece of music. Many applications are imaginable that work only if a mark can be securely detected by the public.

1.1.1 Public Key Watermarking

Such problems could be theoretically avoided by a watermarking algorithm analogous to public key cryptography. Each user has a private key to embed a watermark; a third person can perform the watermark detection using the corresponding public key. Informally, any practical public key watermarking scheme should fulfill the following requirements [13]:

- **Robustness.** The embedding process should be robust; i.e., it should not be possible to remove a watermark without rendering the data useless. Ideally, the public detection procedure should not impair the robustness of the underlying embedding mechanism.
- **Asymmetry.** Knowledge of the public key does not enable an attacker to remove a private watermark; more specifically, the public key must not reveal the location of the private watermark in the digital object.
- **Feasibility.** Both embedding and detection must be computationally feasible.
- **Security.** It must be computationally infeasible to deduce the private key from the public key.
- **Authenticity.** It must not be possible to use the public key to insert a watermark in a digital object (or use the key in protocol attacks).

Unfortunately, such schemes seem to be difficult to engineer, as the following example illustrates. Hartung and Girod [25] presented an extension to their watermarking system (see Section 2.4.1), in which a mark is inserted by a private key but where the presence of the watermark can be checked using a different (public) key. Basically, the private key consists of the pseudorandom sequence p_i. By making only parts of the sequence p_i public and replacing all other bits by a random sequence, they obtain a "public" key p_i^p. On the average, every n-th coefficient is taken from the original sequence:

$$\begin{cases} p_i^p = p_i & \text{with probability } 1/n \\ p_i^p \leftarrow_R \{-1, 1\} & \text{with probability } 1 - 1/n, \end{cases}$$

where $\leftarrow_R \{-1, 1\}$ denotes a random drawing from the set $\{-1, 1\}$. Using this public key, a watermark can be detected in the same manner as indicated above, where p_i^p is used as an replacement for p_i. Due to the redundant embedding of the watermark bits, the watermark can be successfully retrieved.

It is easy to see that the scheme fails on the public watermarking criterion, as the public portion of the key can be removed in the same manner as the complete watermark in the symmetric case: the public watermark $p_i^p \alpha b_i$ is subtracted from the watermarked video. Although the secret watermark could still be successfully detected with the whole key p_i, the benefits of the public detection are lost. After an attack, the watermark owner could construct a new public key using sequence elements not yet revealed. However, this mark is susceptible to the same attack. There is also a possibility of a protocol attack, showing that the system also fails the authenticity requirement, as defined above. An attacker can take the public sequence p_i^p and insert a fake watermark into a different object (which could also be verified with the public key p_i^p).

1.1.2 Asymmetric Watermarking

The ideal paradigm of public watermarking has however lead to a large variety of watermarking schemes that can be qualified as asymmetric schemes. Such schemes have the property that *the set of keys that are used for the embedding and the detection of the watermark is different*, even though they do not necessarily meet all requirements of public key watermarking as mentioned in Section 1.1.1. For example, Hartung and Girod's extended scheme can be considered as asymmetric, because the embedding key p_i and the detection key p_i^p are not identical.

Another property of asymmetric watermarking is the concept of *renewability* defined by Furon *et. al.* [21]:

- If the secret watermark is estimated and erased it is still possible to generate another secret watermark that can be detected with the public detection key.

This property allows to embed different secret watermarks on different documents that share the same public detection key. Hence, if one secret watermark is revealed, contents that is marked with a different secret watermark is still protected.

It is also important to note that there exist asymmetric schemes that have the dual property of the previous one (for example, the scheme by Hartung and Girod satisfies this):

- If the public watermark is estimated and erased it is possible to design a watermark detector that will reveal the presence of the secret watermark.

This last property is certainly not a requirement for asymmetric watermarking schemes but may be convenient in real life applications.

1.2 Asymmetric Versus Zero-Knowledge Watermarking

In order to construct watermarking schemes that avoid the disclosure of a secret detection key that potentially compromises the security of an application, two principal approaches can be found in the literature:

- Truly *asymmetric watermarking schemes* use two different keys for watermark embedding and detection on the signal-processing level. Among them are systems that use properties of Legendre sequences [36], "one-way signal processing" techniques [16] or eigenvectors of linear transforms [17]. Chapter 2 discusses these constructions in detail.
- In contrast to asymmetric schemes, where the detector is designed to use a different key, zero-knowledge watermarking schemes use a standard watermark detection algorithm and a cryptographic zero-knowledge proof that is wrapped around the watermark detector. The idea was first introduced by Gopalakrishnan *et. al.* [24] and later refined by Craver [12], Craver and Katzenbeisser [13, 14] and Adelsbach and Sadeghi [4]. Constructions for zero-knowledge watermark detectors will be described in Chapter 3.

Chapter 2

Asymmetric Watermarking

2.1 Asymmetric Watermarking Using Matrix Products

The aim of this chapter is to provide a critical review of the existing asymmetric watermarking techniques, thereby pointing out possible future research directions. In the first three sections we present asymmetric schemes that use a *quadratic detection* criterion; the fourth section describes *linear* detection schemes. The last section provides a critical review of the presented schemes and outlines future directions for asymmetric watermarking.

2.1.1 Key Independent Watermark Detection

In 1999 van Schyndel, Tirkel and Svalbe [36] proposed an algorithm that is able to verify the presence of a watermark in a digital document without knowing both the watermarking key and the hidden watermark. Their method is based on invariance properties of Legendre sequences with respect to the Discrete Fourier Transform (DFT). In particular, the DFT of a Legendre sequence $\mathbf{l}$ is:

$$\mathbf{L} = DFT\{\mathbf{l}\} = L_1\mathbf{l}^*.$$

That is, the DFT of $\mathbf{l}$ is equal to the conjugate Legendre sequence $\mathbf{l}^*$ up to a constant factor L_1, which equals the first component of the Fourier transform. Hence, they exploit the fact that the auto-correlation values of a Legendre sequence and the cross-correlation values between the sequence itself and its conjugate DFT only differ by a scale factor.

Using this idea, the embedding process consists of modifying the host pixels (or some transformed coefficients) by means of the values of the Legendre sequence. For example, the Legendre sequence may be simply added to the host pixels. During the detection step the algorithm computes the cross-correlation between the received signal $\mathbf{r}$ (i.e., the possibly watermarked content) and its conjugate Fourier transform $\mathbf{R}^*$:

$$c = \frac{\mathbf{r}^T\mathbf{R}^*}{N},$$

where N is the length of $\mathbf{r}$. A watermark is assumed to be present, if this correlation value exceeds a constant threshold.

In order to apply the algorithm to images, the authors proposed to extend the Legendre sequence to a two-dimensional Legendre array by directly multiplying row and column sequences to form a product array. Such an array can then be used for watermark embedding. For simplicity it is also possible to embed in an image a one-dimensional Legendre sequence by scanning the image row-by-row.

2.1.2 Public Key Watermarking by Eigenvectors of Linear Transforms

By relying on the method described in the previous section, Eggers, Su and Girod [17] constructed an asymmetric scheme (called *eigenvector watermarking*). The authors followed the main idea of the previous algorithm (i.e., the invariance property of Legendre sequences under the DFT), but looked at different sequences and transforms with similar properties.

In particular, they proposed to adopt a watermark $\mathbf{w}$ that is an eigenvector of a linear transform matrix $\mathbf{G}$,

$$\mathbf{G}\mathbf{w} = \lambda_0 \mathbf{w}.$$

During the embedding step, the watermark $\mathbf{w}$ is added to the host signal. Watermark detection can again be performed without knowledge of the watermark by computing the correlation between the received signal $\mathbf{r}$ (i.e., the possibly watermarked content) and its transformed version $\mathbf{G}\mathbf{r}$:

$$c = \frac{\mathbf{r}^T \mathbf{G}\mathbf{r}}{N}.$$

The transform matrix should be chosen in order to achieve a good insensitivity of the detector to the host signal and a good robustness and security against malicious attacks. Furthermore, the efficiency of the watermark embedder and detector must be considered. There are two factors that influence the efficiency: the computational complexity of the transform and the existence of a compact representation for the matrix $\mathbf{G}$.

The correlation c is a sum of two contributions, one related to the host signal $\mathbf{x}$ and one related to the watermark $\mathbf{w}$. For a reliable detection result, the interference from the host signal should be negligible—even for a high watermark embedding strength, i.e., for a high value of the Data to Watermark Ratio (DWR). The authors show that the matrix $\mathbf{G}$ should be chosen such that:

$$E\left\{\frac{\mathbf{x}^T \mathbf{G}\mathbf{x}}{N}\right\} \approx 0 \qquad \text{and} \qquad Var\left\{\frac{\mathbf{x}^T \mathbf{G}\mathbf{x}}{N}\right\} \propto \frac{1}{N};$$

this can be achieved if $\mathbf{G}\mathbf{x}$ and $\mathbf{x}$ are uncorrelated.

Regarding robustness, comparing the performance of the proposed public approach with a symmetric scheme shows that in order to achieve approximately the same detection perfor-

mance, the watermark length in the public scheme has to be increased by a factor of DWR[2]. This is a very demanding request if we consider that DWR $\gg 1$ [1].

From a security point of view, Eggers *et. al.* analyzed a possible attack which consists in an exhaustive search of the embedded watermark $\mathbf{w}$. One promising attempt for an attacker is to compute the eigenvalues λ_i of $\mathbf{G}$ and search for the corresponding eigenvectors. If the geometrical multiplicity of the eigenvalue λ_0 is equal to one, then the corresponding eigenvector is unique (i.e., equals $\mathbf{w}$) and may be easily found. To avoid such an attack, the eigenvalue related to the eigenvector $\mathbf{w}$ should have a geometrical multiplicity $\gg 1$. In this case, the corresponding eigenvectors are not uniquely defined and the attacker must do an exhaustive search in a space that increases exponentially with the multiplicity of the eigenvalue.

Another attack against the watermark security consists in confusing the public detector by adding an appropriate sequence $\mathbf{z}$ that is orthogonal to $\mathbf{w}$ to the watermarked content. In particular, let us assume that $\mathbf{z}$ is an eigenvector of $\mathbf{G}$ corresponding to the eigenvalue $-\beta\lambda_0$, with $\beta > 0$ and λ_0 being eigenvalue of $\mathbf{w}$. We have:

$$\mathbf{G}\mathbf{z} = -\beta\lambda_0\mathbf{z}.$$

By adding the scaled sequence $\mathbf{z}/\beta$, the watermark detector will measure zero correlation. Of course, the attacker must consider the quality degradation depending from the addition of $\mathbf{z}$.

A special case of *eigenvector watermarking* uses the Fourier transform as transformation matrix: $\mathbf{G} = \mathbf{G}_{DFT}$. The benefit of this choice is twofold: the detection matrix $\mathbf{G}$ has not to be transmitted to the detector and fast algorithms to compute the transform are known. It is clear that, for real signals, this approach is almost the same as that based on Legendre sequences, with $\mathbf{R}$ instead of $\mathbf{R}^*$:

$$c = \frac{\mathbf{r}^T\mathbf{G}_{DFT}\,\mathbf{r}}{N} = \frac{\mathbf{r}^T\mathbf{R}}{N}.$$

The benefit of the eigenvector approach with respect to the Legendre approach is that it permits to overcome the problems due to the small number of Legendre sequences. In fact, there are only $N-2$ Legendre sequences of length N, thus enabling an efficient exhaustive search for watermarks.

Another useful class of transformation matrices are the permutation matrices $\mathbf{G}_{PERM}$. As in the case of $\mathbf{G}_{DFT}$, these matrices have the benefit of a low cost transmission to the detector and of computational efficiency. In fact, $\mathbf{G}_{PERM}$ can be described through few values (for a signal of length N at most $N-1$ integer values are needed); in addition, the permutation transform only consists of re-indexing operations and is thus computationally efficient.

[1] In the above expression a linear version of DWR is used, whereas in most cases a logarithmic scale is used (e.g., DWR is measured in *dB*).

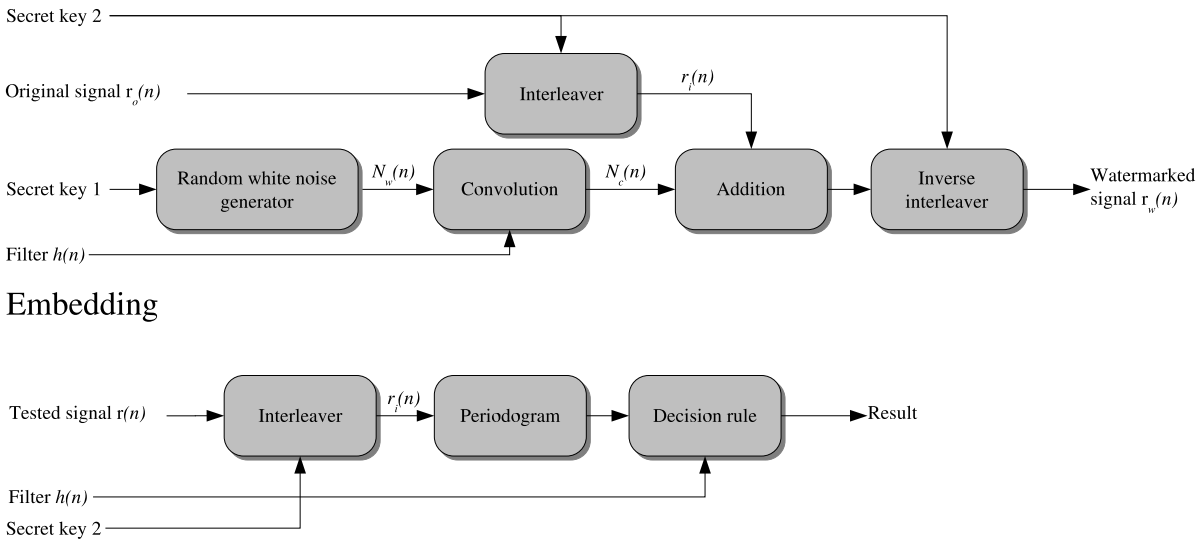


Figure 2.1: Embedding and Detection functions of the asymmetric watermarking scheme presented by Furon and Duhamel.

2.2 Asymmetric Watermarking Using Spectrum Constraints

Furon and Duhamel [16, 20] presented an asymmetric watermarking scheme that modifies the spectrum shape of an interleaved image to perform the embedding of the watermark. The main steps of this scheme are depicted in Figure 2.1[2].

Since the scheme is asymmetric, the set of keys that are needed during the embedding and the detection is different. The embedding of the watermark needs a private key composed of three individual keys:

- a key that enables the generation of white noise $N_w(n), n \in \{0, ..., N-1\}$,
- the coefficients of a convolution filter $h(n)$ that can be convoluted with $N_w(n)$ in order to obtain colored noise $N_c(n)$, and
- another key that acts as parameter of the interleaving function and yields to an interleaved signal[3] $r_i(n)$ from the original signal $r_o(n)$.

Because $N_w(n)$ and $r_i(n)$ can be both considered as white signals, the spectrum after the embedding, done by adding the colored noise $N_c(n)$, will have the same shape as the spectrum of $h(n)$. This fact is used for watermark detection.

[2]This Figure is strictly equivalent to the initial Figure presented by the authors in [20]; for pedagogical purposes we have interleaved the extracted content instead of the colored noise during the embedding process, the detection process remains identical.

[3]The term "signal" means here a component of the media content that can be used to describe it; a signal can be, for example, pixel values, DCT coefficients, wavelet coefficients, etc.

It is important to note that during the detection process, only a set of two keys, the first represented by the coefficients of $h(n)$ and the second represented by the interleaving key, are used. This implies that the original white noise $N_w(n)$, which represents the watermark, cannot easily be removed.

The watermark detection process has to decide if the spectrum of the interleaved signal is similar to the shape of the spectrum of $h(n)$ or not. This is done by calculating an approximation of the likelihood function of the spectrum for each hypothesis; finally both functions are compared with a threshold. For each hypothesis the likelihood $V(r, S_i)$ can be shown to be (using Whittle's theorem):

$$V(r, S_i) = 2N \int_{-1/2}^{1/2} \frac{I(f)}{S_i(f)} + \log S_i(f)\, df,$$

where $S_i(f)$ is the spectrum of each hypothesis (0 for an original content and 1 for a marked content) and $I(f)$ is the periodogram function defined by:

$$I(f) = \left| \sum_{k=0}^{N-1} r[n] e^{2\pi i n f} \right|^2 \qquad \forall f \in \,]-1/2, 1/2].$$

The authors point out that this construction can be adapted to any watermarking scheme that uses Spread Spectrum techniques. In addition, they gave an implementation based on a Direct Sequence Spread Spectrum technique presented by De Rosa *et. al.* [34] using the DFT space for both watermark embedding and detection.

2.3 Unified Approach with Quadratic Detection

In [21] and [19] Furon *et. al.* proposed an unified approach that is able to describe all schemes that have been presented so far. They outline that in the schemes presented by Smith and Dodge[4] [35], Van Schyndel *et. al.* [36], Eggers *et. al.* [17] and Furon and Duhamel [20], the detection function $D(\mathbf{r})$ can be written using a quadratic form $Q()$:

$$D(\mathbf{r}) = \frac{Q(\mathbf{r})}{N} = \frac{\mathbf{r}^T \mathbf{A} \mathbf{r}}{N}.$$

The authors also compare the power of the presented test with the power of a classical spread spectrum test. The power of the test is relative with the deflection coefficient given by

$$\epsilon = \frac{E\{\mathbf{r}|H_1\} - E\{\mathbf{r}|H_0\}}{\sigma_{\mathbf{r}|H_1}},$$

[4]Smith and Dodge proposed a basic asymmetric watermarking scheme that relies on the embedding a periodical random sequence. The detection of the watermark is afterwards done by calculating the cross-correlation of the image (the peaks that are due to periodicity reveal then the presence of the watermark).

where H_1 is the hypothesis when $\mathbf{r}$ corresponds to a watermarked content and H_0 is the hypothesis when $\mathbf{r}$ corresponds to a non watermarked content. For classical spread spectrum schemes, the authors show that

$$\epsilon \sim \frac{\sigma_w}{\sigma_s}\sqrt{N},$$

where σ_s denotes the standard deviation of the original signal.

For asymmetric watermarking schemes based on a quadratic form the expression of the deflection coefficient is given by:

$$\epsilon \sim \frac{\sigma_w^2}{\sigma_s^2}\sqrt{N}.$$

Consequently, because $\sigma_W/\sigma_S < 1$ in watermarking scenarios, the efficiency of asymmetric watermarking methods is smaller than for DSSS watermarking methods. For a classical ratio σ_W^2/σ_S^2 equal to $-20dB$, the length of the random sequence has to be ten times longer for asymmetric watermarking schemes than that for DSSS watermarking schemes to provide similar detection performances.

Nevertheless, in [19] authors also investigate security issues in the cases of detection schemes that use a quadratic form as a detection function, especially its resistance against oracle attacks [28][5]. For classic DSSS watermarking schemes, the attacker has to estimate a watermark of length N. In the asymmetric case, the attacker has to estimate the matrix $\mathbf{A}$ which is represented by a signal of size N^2. The authors note that, even if an attack complexity proportional to $O(N^2)$ is not sufficient to design a secure algorithm, it is better that classical DSSS.

2.4 Linear Asymmetric Watermarking Schemes

Other authors explored the framework of classical spread spectrum watermarking techniques in order to achieve to asymmetry. These schemes rely on the generation of a public key that is a random signal which is partially correlated with the private key. In this approach, the detection of the watermark is not a quadratic but a linear function:

$$D(\mathbf{r}) = \frac{C(\mathbf{r})}{N} = \frac{\mathbf{w_p}^T\mathbf{r}}{N}.$$

It is important to note that, due to the correlation structure of the detector, the public watermark can be easily removed using adequate scaling and subtraction of the public watermark. Several constructions for correlation-based asymmetric watermarking schemes are reviewed below.

[5]The oracle attack is an attack where the attacker has black-box access to a watermark detector: the attacker has the possibility to feed the detector with arbitrarily chosen content and observe the detection results, but has no access to the internal structure of the detector.

2.4.1 Partial Key Embedding System

Hartung and Girod [25] were the first to design an asymmetric watermarking scheme based on correlation. This scheme, already presented in Chapter 1, relies on the addition of a very large random sequence that depends on a private key. Each public key is thereafter generated by taking one part of the initial samples of the private key. The size of the public watermark is chosen in such a way that the number of samples is sufficient to guarantee the detection of the public watermark but also allows the detection of the secret watermark by subtracting the private sequence (e.g., the private key).

2.4.2 Transformed-Key Watermarking System

Choi *et. al.* [9] proposed another correlation-based asymmetric watermarking scheme which requires a linear transform (defined by a matrix $\mathbf{A}$) to generate both the private key and the public key. Using a random secret vector $\mathbf{u}$, the secret key and public keys are respectively given by $\mathbf{Au}$ and $\mathbf{A}^{-T}\mathbf{u}$.

The embedding process adds a weighted private watermark $\mathbf{w_{pr}} = \gamma_{pr}\mathbf{Au}$ to the host signal $\mathbf{x}$:

$$\mathbf{y} = \mathbf{x} + \alpha\mathbf{w_{pr}} = \mathbf{x} + \alpha\gamma_{pr}\mathbf{Au}.$$

The detection is performed by correlating the received signal $\mathbf{r}$ with the public watermark $\mathbf{w_{pu}} = \gamma_{pu}\mathbf{A}^{-T}\mathbf{u}$:

$$\mathbf{w_{pu}}^T\mathbf{r} = \gamma_{pu}\mathbf{u}^T\mathbf{A}^{-1}\mathbf{x} + \gamma_{pu}\mathbf{u}^T\mathbf{A}^{-1}\alpha\gamma_{pr}\mathbf{Au} = \gamma_{pu}\mathbf{u}^T\mathbf{A}^{-1}\mathbf{x} + \alpha\gamma_{pu}\gamma_{pr}\mathbf{u}^T\mathbf{u}.$$

We can note that the matrix $\mathbf{A}$ acts as a scrambling function that generates the private embedded mark $\mathbf{w_{pr}}$ from $\mathbf{u}$. The matrix $\mathbf{A}^{-T}$ is used to cancel the effect of $\mathbf{A}$ during the detection process without revealing $\mathbf{u}$.

It is important to point out that this scheme has several important drawbacks:

- As other schemes of this category, the public watermark can be trivially removed just by subtracting a scaled version of $\mathbf{w_{pu}}$.
- The matrix $\mathbf{A}$ and the vector $\mathbf{u}$ have to be carefully chosen in such a way that their cross correlation is not too big to prevent the removing of the private watermark.
- If a large set of privates key is used it is possible to estimate the matrix $\mathbf{AA}^T$ and consequently to remove the private key.

2.4.3 Private Keys Generation Using Phase-shift-transforms

Kim *et. al.* [29] have developed another public key generation scheme that provides partial correlation with the secret watermark. Contrary to previous correlation-based schemes, a set of private watermarks is generated for one public watermark. The authors point out that

such a technique can be useful to allow multiple detection of a same public watermarking without having the possibility to estimate the private watermark using several watermarked images. The construction of private watermarks is done using the phase-shift-transform. The public watermark $w_{pu}(n)$, chosen as a random sequence, is transformed in the DFT domain, yielding $W_{pu}(k)$. Then the frequency components of one secret key are defined by $W_{pr}(k) = W_{pu}(k)e^{j^{\Phi}(k)}$, where $\Phi(k)$ is a binary random sequence with two possible values $-\Phi_0$ and Φ_0. This operation was named phase-shift-transform by the authors. The normalized correlation between w_{pu} and w_{pr} is given by $\cos(\Phi_0)$. Consequently, the parameter Φ_0 enables to choose the degree of correlation between the public and the secret watermark. The authors choose $\Phi_0 = 0.5$ to prevent the loss of the private detection by removing the public key.

2.5 A Critical View of Asymmetric Watermarking: Misconceptions and Potentials

In this section we give a critical overview of the asymmetric watermarking algorithms proposed so far. More specifically, by slightly changing the point of view of our analysis, we will see that virtually all the systems proposed so far failed to use asymmetry to increase security. This is evident when the informed embedding paradigm is taken into account.

2.5.1 Early Algorithms: the Wrong Approach

For sake of simplicity, in the following, we will focus on watermark detection, the extension to multibit watermarking being straightforward. Let us indicate by $\mathbf{x} = (x_1 \dots x_n)$ the row vector with the original, to-be-marked features, let $\mathbf{y} = (y_1 \dots y_n)$ be the marked feature vector, and $\mathcal{E}$, $\mathcal{D}$ denote, respectively, the embedding and detection function. We clearly have:

$$\mathbf{y} = \mathcal{E}(\mathbf{x}, K_e), \tag{2.1}$$

$$\mathcal{D}(\mathbf{y}, K_d) = \text{yes/no}, \tag{2.2}$$

where K_e and K_d are the embedding and detection keys respectively. The definition of $\mathcal{D}$ and the associated detection key K_d automatically partitions the feature space into two regions, let us call them the watermarked region I_w and the non-watermarked region I_0. Given this basic definition of the watermarking process, the task of the embedding function $\mathcal{E}$ can be simply described as: *given the to-be-marked vector* $\mathbf{x}$*, find a point in* I_w *which is close enough to* $\mathbf{x}$ *and far enough from the border of* I_w *so to achieve a desired level of robustness.*

Note that the term *close enough* must be understood in a perceptual sense, and that the definition of robustness is purposely vague, being its role marginal in this context. The above definition of the watermarking problem reflects a typical informed-embedding point of view, where the watermarking signal, let us call it $\mathbf{w}$, that needs to be added to $\mathbf{x}$ in order to move it into I_w may depend on $\mathbf{x}$ itself, and is not part of the embedding key K_e. Note that this was not the case with blind-embedding methods, e.g., with spread spectrum watermarking, where the watermarking signal was considered to be part of K_e and, hence, it did not depend on $\mathbf{x}$.

In spite of the above observations, most of the asymmetric algorithms proposed so far rely on the assumption that the watermarking signal is part of the embedding key, and achieve asymmetry by avoiding that the detector uses it to decide whether $\mathbf{y}$ belongs to I_w or not. Let us consider, for example, the very simple asymmetric watermarking scheme developed by Smith and Dodge in 1999 [35]. The feature vector $\mathbf{x}$ is split into two equal parts and to each part the same pseudorandom signal is added:

$$y_i = x_i + \gamma w_i, \tag{2.3}$$

$$y_{i+n/2} = x_{i+n/2} + \gamma w_i, \tag{2.4}$$

for $1 \leq i \leq n/2$. The detector simply computes the correlation between the first and the second part of the watermarked feature vector, i.e.,

$$c = \frac{2}{n} \sum_{i=1}^{n/2} y_i y_{i+n/2}, \tag{2.5}$$

and compares it against a detection threshold. In order to consider the above scheme as an asymmetric algorithm, it is necessary that the watermarking signal $\mathbf{w}$ is seen as the embedding key, whereas no detection key is needed. If we follow the informed embedding point of view, however, the choice of the particular $\mathbf{w}$ to be added to $\mathbf{x}$ has not to be considered as part of K_e, since it is better seen as an output (or to better say a side-output) of $\mathcal{E}$, rather than one of its inputs. On the contrary, the keys K_e and K_d are only intended to describe the watermarked region I_w. We could also use the above argument to state that in the system proposed by Smith and Dodge [35] the embedding and detection keys are basically empty sets.

As we have seen previously, in more sophisticated asymmetric systems, the watermarked region is defined by means of a quadratic form, so that

$$\mathcal{D}(\mathbf{y}, K_d) = \text{yes} \quad \text{iff} \quad \frac{\mathbf{y}^T A \mathbf{y}}{n} > T, \tag{2.6}$$

where the square matrix A is needed both at the embedder and the detector, and hence it plays both the role of the embedding and detection keys $K_e = K_d = \mathbf{A}$. For the simple scheme described previously we would have

$$\mathbf{A} = 2 \begin{bmatrix} 0_{n/2} & I_{n/2} \\ I_{n/2} & 0_{n/2}. \end{bmatrix}. \tag{2.7}$$

Note that, unlike required by the asymmetric strategy, $K_e = K_d$, the ignorance of the watermarking signal by $\mathcal{D}$ being irrelevant. Then why are the schemes described in the previous sections more secure than classical spread spectrum watermarking? Because the shape of the watermarking region is more complex (it needs more parameters to be described), hence making the implementation of the the sensitivity attack (followed by a closest point attack) more difficult (complex)[6].

[6] Under this perspective the natural way of extending the analysis in [21], is to further increase the complexity of the watermarked region, e.g. by using higher order functions of $\mathbf{x}$ [27].

2.5.2 Perspectives for Future Research

The wrong approach to the problem of asymmetric watermarking, that characterized early algorithms, may lead one to think that asymmetric watermarking is not the right answer to the security threats set by the sensitivity and the closest point attacks in a public detection framework. However this is not necessarily true. In order to understand how asymmetric watermarking may improve the security of watermarking systems, let us consider again the task of the embedder and let us compare it to that of the attacker. Given a point $\mathbf{x}$ in I_0 (if $\mathbf{x} \in I_w$, then $\mathcal{E}$ may let $\mathbf{y} = \mathbf{x}$), it is the embedder's goal to find a point within I_w which is close enough to $\mathbf{x}$. What about the attacker, then? Given a point $\mathbf{y}$ in I_w, the attacker must find a point within I_0 which is close enough to $\mathbf{y}$. It is readily seen that the attacker shares essential the same (we could say the dual) goal of the embedder. Why should attacker's work be more difficult than that of the embedder? Possibly because the embedder exactly knows I_w while the attacker does not. This corresponds to the symmetric approach where $K_e = K_d = I_w$ (note that the detector surely knows I_w since otherwise it could not verify whereas $\mathbf{y}$ lies within it or not). As we know, this approach is effective as long as the attacker can not estimate K_d, however in the public detection scenario, this hypothesis does not hold. A possible solution is to continue adopting a symmetric approach and make the estimation of K_d (the shape of I_w) as difficult as possible (as it is essentially done by the *asymmetric* algorithms proposed so far). Interestingly, the similarity between the embedder's and attacker's goal points out a problem of this approach: by complicating the shape of I_w, we certainly increase the security of the system, however we also make the embedder's task more difficult.

An alternative solution is to use asymmetric watermarking. A first possibility in this direction, is that the embedder and the detector use two different watermarked regions $I_{w,e}$ and $I_{w,d}$, with $I_{w,e} \subset I_{w,d}$. If the shape of $I_{w,d}$ is much more complicated than that of $I_{w,e}$, then it may be difficult for the attacker to estimate it, and, once the estimation is known, to apply the closest point attack (this is not the case for the embedder since $\mathcal{E}$ relies on the simpler region $I_{w,e}$). A proposal in this direction has been made in [31], where by starting from a simple-shaped $I_{w,e}$, a watermarked region $I_{w,d}$ with a much more complicated shape is built by relying on fractal theory. The problem with this approach is that $I_{w,e} - I_{w,d}$ must be as small a set as possible, so that the false detection probability is not increased too much. This requirement, in turn, makes it possible for the attacker to use a rough easy-to-compute estimate of $I_{w,d}$ to perform his attack.

A second solution is to use the same watermarked region, but provide the embedder and the detector with two different descriptions of it. For example, the detector could be provided with an implicit non-invertible, description of I_w while an explicit description is given to the embedder. As far as we know no algorithm has been developed so far in this direction.

As a last resort, the set I_w could be built in such a way that it is easy to *enter it*, but very difficult to *exit from it*. This would be a perfect solution, since the need to keep the shape of I_w secret would disappear, security being granted by the nature itself of the embedding and the attack problems. This approach, where nothing has to be kept secret, is sometimes referred to as *open cards* or *open hands* watermarking [6]. Though interesting, the viability of such an approach is rather questionable. Some possible directions to build a watermarked region matching the requirements of the open cards scenario are given in [32].

Chapter 3

Zero-Knowledge Watermarking

3.1 Zero-Knowledge Watermark Detection Protocols

In contrast to asymmetric schemes, where the detector is designed to use a different key, zero-knowledge watermarking schemes use a standard watermark detection algorithm and a cryptographic zero-knowledge proof that is wrapped around the watermark detector. The idea was first introduced by Gopalakrishnan et al. [24], who describe a protocol that allows an RSA-encrypted watermark to be detected in RSA-encrypted content. However, the protocol was not truly zero-knowledge. Subsequent research by Craver [12], Craver and Katzenbeisser [13, 14] and Adelsbach and Sadeghi [4] concentrated on the construction of cryptographic zero-knowledge proofs for watermark detectors. An overview and summary of zero-knowledge watermark detection can be found in [1, 2].

The goal of zero-knowledge watermark detection is to prove the presence of a specific watermark in a digital object *without compromising the security of this watermark*. To achieve this, all security-critical parameters, i.e., the watermark and the detection key, are *encoded* and watermark detection is performed on the encoded parameters, without removing the encoding. Such protocols ideally fulfill the following two requirements:

1. **Inputs conceal watermark and key.** The encoded inputs do not reveal any information about the watermark and the detection key.
2. **Protocol is zero-knowledge.** A run of the protocol does not disclose any information *in addition* to the inputs of the protocol and the binary watermark detection result.

These properties guarantee that a watermark stays as secure as if only the detection result has been revealed. Zero-knowledge watermark detection can improve the security of many applications which rely on symmetric watermarking schemes, and can reduce the necessary trust in certain parties or devices.

3.1.1 Interactive Proof Systems

For a detailed introduction to interactive proof systems and zero-knowledge proofs, we refer to [33, 22].

Formally, a zero-knowledge proof is an interactive proof system, which can be described as a two-party protocol with output between two entities $\mathcal{P}$ and $\mathcal{V}$. $\mathcal{P}$ is called "prover", whereas $\mathcal{V}$ is called "verifier". The prover's task is to prove a statement to the verifier; this statement is encoded in the *common input* of the protocol. The output is either $\top$ or $\bot$, indicating whether the verifier accepts or rejects the statement. Both parties have access to an *auxiliary input*, encoding secret information.

The fundamental security properties of an interactive proof system are *completeness* and *soundness*:

- **Completeness.** A correct prover $\mathcal{P}$ can prove all correct statements to a correct verifier $\mathcal{V}$.
- **Soundness.** A cheating prover $\mathcal{P}^*$ cannot prove a wrong statement to an honest verifier. That is, a verification procedure cannot be faked such that a honest $\mathcal{V}$ accepts false statements. Note that this property is usually probabilistic, i.e., there may be a tolerated success probability for a cheating prover.

In the cryptographic literature, two main types of proof systems can be identified:

- **Proof of language membership** for a fixed language L. Here, the prover $\mathcal{P}$ wants to convince the verifier $\mathcal{V}$ that a string x, called common input, available to both parties, satisfies indeed $x \in L$. (Note that trivially each language $L \in \mathbf{NP}$ has an interactive proof system).
- **Proof of knowledge** for a fixed relation R. Again, both $\mathcal{P}$ and $\mathcal{V}$ share a common input x. In a proof of knowledge, $\mathcal{P}$ wants to prove to $\mathcal{V}$ that he "knows" a string Aux, called *witness*, such that $(x, Aux) \in R$.

In the rest of this work, we will denote with Γ a set of (numeric) security parameters, describing, among others, the degree of confidence in the proof system or the strength of the hiding property. Furthermore, GENERATE will denote the generating algorithm. On input Γ, GENERATE outputs a pair (x, Aux), where x is the common input and Aux denotes the corresponding auxiliary input of the prover $\mathcal{P}$.

An interactive prove protocol is a two-party cryptographic protocol between $\mathcal{P}$ and $\mathcal{V}$, where the common input is given by x and Γ, and $\mathcal{P}$'s private input by Aux. During the protocol, $\mathcal{P}$ and $\mathcal{V}$ exchange messages and at the end, $\mathcal{V}$ outputs either $\top$ or $\bot$, indicating whether $\mathcal{V}$ accepts or rejects the proof.

Most proof protocols have a challenge-response form. Given the common input, the protocol consists of three moves: the prover $\mathcal{P}$ starts by sending a message to $\mathcal{V}$, who in turn responds by sending a challenge to $\mathcal{P}$; in the last step, $\mathcal{P}$ sends his answer back to $\mathcal{V}$, who verifies its correctness.

Formally, an interactive proof system for language membership is defined as follows:

Definition 1 *Let L be a language, Γ be the set of security parameters, and $\gamma \in \Gamma$ a security parameter. Further, let* GENERATE *be a generating algorithm, and $\mathcal{P}$ and $\mathcal{V}$ be interactive algorithms. An interactive proof system for language membership providing information-theoretical soundness over L is an interactive cryptographic protocol between $\mathcal{P}$ and $\mathcal{V}$ such that*

1. ***Correct generation.*** *For all security parameters Γ and all tuples $(x, Aux) \leftarrow$ GENERATE(Γ), $x \in L$ holds, i.e.,* GENERATE *generates only elements of the language L.*

2. ***Completeness.*** *For all parameters Γ and all $(x, Aux) \leftarrow$ GENERATE(Γ), a correct prover can always convince a correct verifier $\mathcal{V}$ of $x \in L$, i.e.,*

$$\mathbf{P}[\mathcal{V}_{\mathcal{P},Aux}(\Gamma, x) = \top] = 1.$$

3. ***Soundness.*** *For all interactive algorithms $\mathcal{P}^*$, for all valid parameters Γ, for all $x \notin L$ and for all $Aux \in \{0,1\}^*$,*

$$\mathbf{P}[\mathcal{V}_{\mathcal{P}^*,Aux}(\Gamma, x) = \top] \leq 2^{-\gamma}.$$

Here, we denote with $\mathcal{V}_{\mathcal{P},Aux}$ the probabilistic algorithm $\mathcal{V}$ when interacting with the prover $\mathcal{P}$, whose private input is Aux. Informally, the soundness assures that a cheating prover cannot incorrectly convince a correct verifier of $x \in L$. Note that no restriction is placed on the computational power of the verifier; we therefore speak of *unconditional soundness*. Alternatively, one may also consider only provers whose computational power is restricted, namely bound to polynomial computations.

A formal definition of proofs of knowledge can be found in [22] and [2].

3.1.2 Zero-Knowledge Property

Informally, a proof system is said to be zero-knowledge, if the system reveals "no knowledge" to the verifier, except the fact that the assertion is valid. In other words, the verifier should gain "no new knowledge" from the conversation with the prover during a protocol run that he cannot readily compute from the inputs of the protocol alone. More formally, the verifier gains no new knowledge from the protocol run, if he could easily compute his *view* of the proof by only having the common input x and no interaction with the prover. The view consists of the messages the verifier exchanges with the prover, its states and the content of its random tape.

The zero-knowledge property is a security requirement defined to protect provers and should be guaranteed as long as the provers follow the protocol. Thus, zero-knowledge considers only honest provers whereas the verifier is in general considered to be an adversary $\mathcal{V}^*$ who wants to extract knowledge from the prover. In contrast to an honest verifier, $\mathcal{V}^*$ may have an *auxiliary input* $Aux_{\mathcal{V}^*}$. This input can be interpreted as the prior knowledge of the

verifier which it may have obtained during other protocol-runs with the prover (in which the prover may have used the same auxiliary input).

The zero-knowledge property requires that whatever can be efficiently computed from x and $Aux_{\mathcal{V}^*}$ after completing the interaction with the prover on any x, can be computed by $\mathcal{V}^*$ from x and $Aux_{\mathcal{V}^*}$ without interaction with the prover.

To prove this property, one usually shows the existence of an algorithm called *simulator* $\textsc{Sim}_{\mathcal{V}^*}$ which, given the inputs of the verifier (i.e., the common input x and the auxiliary input $Aux_{\mathcal{V}^*}$), can compute the view of the verifier. Note that cheating verifiers $\mathcal{V}^*$ might deviate from the protocol specification, and might produce a view different from that of the honest verifier. Hence, we are required to give a simulator $\textsc{Sim}_{\mathcal{V}^*}$ for *every* $\mathcal{V}^*$. In the following, we consider only black-box simulation, i.e., there is a *universal simulator* which, given any $\mathcal{V}^*$ as a black-box and $\mathcal{V}^*$'s inputs, simulates the view of $\mathcal{V}^*$ step-by-step, where $\textsc{Sim}_{\mathcal{V}^*}$ is given the capability (privilege) to reset $\mathcal{V}^*$'s state. We will allow the simulator to fail with a certain bounded probability; in this case, $\textsc{Sim}_{\mathcal{V}^*}$ outputs some special symbol $\perp$.

The view of the verifier $\textsc{View}(\mathcal{V}^*, \mathcal{P})$ is a random variable defined by the run of the proof protocol with the honest prover $\mathcal{P}$. The view simulated by the simulator $\textsc{Sim}_{\mathcal{V}^*}$ is denoted by $\textsc{Sim}_{\mathcal{V}^*}(x, \Gamma, Aux_{\mathcal{V}^*})$.

Definition 2 *Let $(\mathcal{P}, \mathcal{V})$ be an interactive proof system. The proof system $(\mathcal{P}, \mathcal{V})$ is called* perfect auxiliary zero-knowledge*, if for all probabilistic interactive algorithms $\mathcal{V}^*$, there exists a (non-interactive) probabilistic algorithm (called simulator) $\textsc{Sim}_{\mathcal{V}^*}$ such that for all parameters Γ, for all $(x, Aux) \leftarrow \textsc{Generate}(\Gamma)$ and for all $Aux_{\mathcal{V}^*} \in \{0,1\}^*$ the following conditions hold:*

- *On input x, $\textsc{Sim}_{\mathcal{V}^*}$ outputs the symbol $\perp$ with probability at most $1/2$,*
- *The two probability distributions of $\textsc{View}(\mathcal{V}^*, \mathcal{P})$ and $\textsc{Sim}_{\mathcal{V}^*_\top}(x, \Gamma, Aux_{\mathcal{V}^*})$ are identical, where the latter denotes the random variable $\textsc{Sim}_{\mathcal{V}^*}(x, \Gamma, Aux_{\mathcal{V}^*})$, conditioned on values other than $\perp$.*

Variations of this definitions are possible. A proof system is called statistically zero-knowledge if the two distributions $\textsc{View}(\mathcal{V}^*, \mathcal{P})$ and $\textsc{Sim}_{\mathcal{V}^*_\top}(x, \Gamma, Aux_{\mathcal{V}^*})$ are statistically indistinguishable; the proof system is called computationally zero-knowledge if they are computationally indistinguishable [22].

It can be shown that the *sequential composition* of auxiliary zero-knowledge proofs is also zero-knowledge, i.e., if subsequent zero-knowledge protocols are performed, then the composed protocol is also zero-knowledge (see [23] and [22]). The same result holds for the sequential composition of polynomially many proofs. This result is very fundamental and useful when designing zero-knowledge protocols. One usually constructs a protocol, called *atomic proof*, for proving a certain assertion. However, the atomic proof normally does not prove the claim completely, especially there may be a certain success probability for a cheating prover to convince the verifier. To handle this, the atomic proof is repeated until a certain degree of confidence is achieved. Now, the sequential composition lemma guarantees that if the atomic proof is zero-knowledge, so is also the proof which results from the repetitions

of the atomic proof. A further application of this composition lemma is that complex zero-knowledge proofs can be assembled from several zero-knowledge proofs, while maintaining the overall zero-knowledge property.

3.1.3 Design of Zero-Knowledge Watermark Detectors

A zero-knowledge watermarking scheme is an interactive proof system between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$; the task of the prover is to convince the verifier that a certain watermark is present in a digital object. The protocol is designed as follows:

- **Common input.** The common input of $\mathcal{P}$ and $\mathcal{V}$ consists of a (possibly modified) digital object $\overline{O}$ and encodings of the watermark and the detection key as well as certain public parameters. This encoding must perfectly "hide" the watermark and the key (note that if these parameters were input as plain text, even the standard watermark detector would be zero-knowledge, since no *new*, i.e., hard to compute, knowledge is gained from the detector's output).
- **Auxiliary input.** The prover's auxiliary input contains some secret information about the common input, which might be the unmarked object or secret keys controlling the encoding.
- **Proof statement.** The statement proved is either a proof of language membership or a proof of knowledge. In the former case, the membership of the common input x in a language L must imply (by the construction of the protocol) that a watermark is detectable. In the latter case, knowledge of a witness must imply successful watermark detection.

The security guarantees are the following:

- **Zero-knowledge property.** The proof protocol and its outputs disclose no additional knowledge on the watermark, the detection key and the original object, i.e., the proof is zero-knowledge.
- **Completeness.** The completeness of the prove procedure guarantees that watermark detection "works", i.e., that any honest prover can prove the presence of a watermark to a correct verifier.
- **Soundness.** The soundness of the prove procedure assures that a cheating prover cannot trick a honest verifier into accepting that a watermark is detectable, although the underlying watermark detector would fail to report its presence.

Remark on Ambiguity Attacks Note that the zero-knowledge property is a property of the detector. Whenever a watermark is detectable in the underlying (symmetric) watermarking scheme, the presence of this mark can also be proved in zero-knowledge. The soundness of the prove procedure only assures that a verifier will not accept an encoded watermark, whose presence cannot be detected by the underlying watermark detector. This implies that

the verifier *cannot* distinguish whether a watermark was previously embedded by the prover (or some other party) or whether the detectable mark is a false positive. Although this also holds with standard symmetric watermarking schemes, ambiguity attacks are considerably more difficult to prevent with zero-knowledge watermark detectors. The reason for this is that the watermark cannot be disclosed during the detection procedure; common countermeasures (like the use of a digital signature as part of the watermark) are much more difficult to implement. Similar problems arise when special properties of watermarks (e.g., whether the watermark contains some fixed identity string) must be verified during a protocol run. These problems can be solved in several ways; for an overview of possible implementations we refer to [2, 3].

3.1.4 Comparison of Zero-Knowledge Watermark Detectors

The general characterization of zero-knowledge watermark detection, as given in Section 3.1.3, leaves several degrees of freedom. One can imagine several, more or less reasonable, definitions of zero-knowledge watermark detection derivable from the characterization, each offering different levels of security. These possible definitions can be compared according to the following criteria [2]:

- **Encoding of common inputs.** The encoding of the common inputs must provide sufficient security; if the common input already leaks information about the original object or the watermark, there is no need for a zero-knowledge protocol, as an attacker can readily compute all information from the common inputs to the protocol. Ideally, the encoding should be performed with a statistically hiding bit-commitment scheme.

  Secrecy of this encoding is perhaps the most crucial issue in zero-knowledge watermark detection. In certain applications the secrecy of this encoding is even more important than the zero-knowledge property of the protocol itself, because the common inputs may be publicly available (e.g., in a public database), even if the zero-knowledge watermark detection protocol is not executed at all.

- **Domain covered by common inputs.** Watermark detection generally works on arbitrarily modified documents. The robustness of the procedure assures that watermarks stay detectable, even after heavy modifications. Ideally, a zero-knowledge watermarking scheme covers the same detection inputs as the standard watermark detector. A priori this is not guaranteed, as Definition 1 and 2 only require the completeness, soundness and zero-knowledge properties for *unmodified inputs*. Unfortunately there are zero-knowledge watermark detection schemes, which do not cover the same domain of detection inputs as the underlying watermark detector, and applying them to common inputs which are intentionally modified by an attacker may have strong negative impact on the security guarantees:

  - For common inputs that were *not* computed according to the generating procedure, the completeness property is not guaranteed to hold. This means that watermark detection might not work at all.
  - For common inputs $x \notin L$ (or $x \notin L_R$), the zero-knowledge property does *not* necessarily hold. Some schemes can guarantee the zero-knowledge property *only* if the

common input is restricted in such a way that the detection protocol is performed for the *unmodified watermarked work*, or at least one which was not maliciously modified by the verifier. This is a very strong assumption, since it contradicts the robustness property of watermarking schemes (however, such schemes may be useful in protocols that require watermark detection in unmodified works only).

If a zero-knowledge watermark detection scheme with restricted common inputs is used in a watermarking protocol, the prover must take care that he only participates in protocol-runs for *valid* common inputs $x \in L$ or $x \in L_R$, respectively.

- **Zero-Knowledge Property of the Detection Protocol.** There are certain degrees of freedom in the definition of zero-knowledge (e.g., one may require information-theoretical zero-knowledge or accept the weaker notion of computational zero-knowledge).

  Watermark detection protocols which do not fulfill a cryptographic zero-knowledge property may still conceal most of the security critical information, and only leak a certain amount of information. However, it is difficult to prove an upper bound on the information leaked during each run, which would be desirable to estimate how many runs one can do without getting compromised. In most cases, a lower bound on the information loss can be specified by giving a concrete attack, which recovers partial secret information during each protocol-run.

3.1.5 Early Approaches to Zero-Knowledge Watermarking

Exploiting Ambiguity Attacks

It is possible to construct a protocol that relies on the possibility of performing an *ambiguity attack* [12]. Such attacks attempt to compute a watermark, which has never been embedded in a digital object O', but nevertheless can be detected there. The idea of the scheme in [12] is as follows: The valid watermark WM is concealed among a set of n fake watermarks constructed through ambiguity attacks. Now, the adversary (equipped solely with a watermark detector) cannot decide which of the watermarks is not counterfeit. The prover has to show that there is a valid watermark in this list without revealing its position. Here, a watermark is called valid, if the prover knows its discrete logarithm (w.r.t a specific generator g) in $\mathbb{Z}_p^*$.

The protocol consists of two steps: watermark detection for n watermarks and a zero-knowledge proof of knowledge for the discrete logarithm problem. The detection process is successful, if some watermarks $WM_{j_1}, \ldots, WM_{j_l}$ are still present and the prover $\mathcal{P}$ can convince the verifier $\mathcal{V}$ that he knows the discrete log of at least one of these watermarks. For details, we refer to [12].

Note that during the protocol no attempt is made to "encrypt" the true watermark WM_j. It is just hidden among a large number of "fake" ones. A potential attacker does not know which watermark is genuine and just has the option of removing all watermarks from the marked data. As the fake watermarks contain large parts of the digital data, their removal will result in great distortions. The hope is that such an attack is infeasible due to the poor quality of the resulting data.

The protocol, as outlined above, is *not* zero-knowledge. A dishonest verifier $\mathcal{V}^*$ can try to successively remove the watermarks WM_i until the proof fails. In this case, $\mathcal{V}^*$ knows that he has removed the genuine mark. A possibility for making the protocol zero-knowledge might be to abort the detection protocol in case not *all* watermarks are detectable. However, this change would decrease the robustness of the detection protocol, since removing one watermark (even a fake one) would let the whole detection protocol fail.

RSA Homomorphic Property

A further protocol for zero-knowledge watermark detection has been proposed in [24], as a solution to the *watermarking decision problem*: Given certain stego-data $\overline{O}' = (\overline{O}'_1, \ldots, \overline{O}'_k)$, decide whether an RSA encrypted watermark $E(WM) = (E(wm_1), \ldots, E(wm_k))$ is present in this stego-data. The authors propose a multi-round challenge-response protocol for solving this problem for the blind version of the well-known watermarking scheme of Cox et al. [10]. In each round the prover chooses a random number r, derives a random sequence $\mathcal{B}$ from it by using some one-way (hash)-function, computes a blinded version $\overline{O}'' = \overline{O}' + \mathcal{B}$ of the stego-image and sends its encryption $E(\overline{O}'') = (E(\overline{O}''_1), \ldots, E(\overline{O}''_k))$ to the verifier. Then, the verifier chooses a random bit and, depending on this bit, challenges the prover either to prove that $E(\overline{O}'')$ is correctly blinded (by revealing r) or to prove that the correlation value of $\overline{O}''$ and WM exceeds the detection threshold. The latter is achieved by letting the prover send parts of the correlation $P_i = \overline{O}''_i * wm_i$ to the verifier, who verifies their correctness as follows: the verifier computes $E(P_i)$, i.e., encrypts P_i using the public encryption key, and compares it to $E(\overline{O}''_i) * E(wm_i)$. If P_i was correct, both should be identical due to the homomorphic property of RSA. Being convinced of the correctness of P_i, he can compute the correlation value simply by adding them.

The security argument is as follows: if sufficiently many rounds have been performed, the verifier can be sure that the prover used randomly blinded versions $\overline{O}''$ of the stego-image and that the watermark correlated with $\overline{O}''$. Since the blinding values $\mathcal{B}$ were random they should not correlate with WM and have no effect on the computed correlation values. Hence, in each round the correlation value between $\overline{O}''$ and WM is a good approximation of the correlation value between the actual stego-image $\overline{O}'$ and the watermark WM. However, no real soundness proof has been given for this protocol and it is not zero-knowledge since the verifier obtains a good estimation of the correlation value.

3.2 Computing with Committed Values

In this section, we describe one zero-knowledge watermark detection protocol [5] in detail.

The idea of this protocol is as follows: the common inputs, among others the watermark, are encoded in commitments. During the protocol, $\mathcal{P}$ and $\mathcal{V}$ jointly and verifiably compute the values according to the underlying detection statistic, where all computations are performed on commitments. More concretely, a commitment on the correlation value is computed by (i) exploiting the homomorphic property of the underlying commitment scheme, (ii) applying the existing zero-knowledge protocols for showing relations between committed values (e.g., from [8]), and (iii) using zero-knowledge protocols to prove that the committed correlation

value exceeds the detection threshold (e.g., from [7]).

The protocol builds on an early protocol by [24], but improves its results, since the watermark is statistically hidden in the commitment and the protocol itself can be proven to be a secure zero-knowledge proof.

3.2.1 Building Blocks

The following protocol uses various building blocks from cryptography.

Commitment scheme. The protocol requires commitments with a *homomorphic property*: Let C_{m_1}, C_{m_2} be commitments to arbitrary messages $m_1, m_2 \in \mathcal{M}$ and let $sk^{m_1}_{\text{COM}}, sk^{m_2}_{\text{COM}}$ be the corresponding secret opening information. The homomorphic property allows the committer to compute commitments that he can open to linear combinations of m_1 and m_2 without revealing any additional information about the content of the involved commitments. More concretely

$$\textsc{Open}(C_{m_1} * C_{m_2}, par_{\text{COM}}, m_1 + m_2, sk^{m_1}_{\text{COM}} + sk^{m_2}_{\text{COM}}) = (m_1 + m_2, \top)$$

$$\textsc{Open}((C_{m_1})^a, par_{\text{COM}}, a * m_1, a * sk^{m_1}_{\text{COM}}) = (a * m_1, \top)$$

holds.

We propose to apply the Damgård-Fujisaki integer commitment scheme [15], which is a generalization of the Fujisaki-Okamoto commitment scheme [18]. This commitment scheme is statistically hiding, computationally binding under the root assumption and can commit to any integer [15].[1] A commitment on a message m is computed as $C_m := g^m h^r mod n$, where n is a product of two safe primes, h is a generator of a large subgroup of $\mathbb{Z}^*_n$ and g is a power of h. For the concrete setup of these parameters we refer to [15].

Proving knowledge of opening information. Given a commitment C_a, we need zero-knowledge proofs for proving knowledge of a message a and secret opening information sk^a_{COM}, such that $\textsc{Open}(C_a, par_{\text{COM}}, a, sk^a_{\text{COM}}) = (a, \top)$, i.e., the prover can open C_a. For the commitment schemes mentioned above, such proofs can be found in [18] and [15] respectively. We denote this protocol with

$$\text{POK}(C_a; (a, sk^a_{\text{COM}}) : \textsc{Open}(C_a, par_{\text{COM}}, a, sk^a_{\text{COM}}) = (a, \top)).$$

For commitment schemes similar to that in [18], this protocol is statistically zero-knowledge and computationally sound under the discrete logarithm assumption for the underlying group.

[1] Loosely speaking, statistically hiding means that the commitment perfectly hides its content, and computationally binding under the root assumption means that if an adversary algorithm manages to break the binding property then it will be able to break a cryptographic assumption, which is commonly believed to be hard. For the commitments we are concerned with this assumption is called the generalized root assumption (see [15]).

Proving relations between committed numbers. During the protocol, $\mathcal{P}$ must be able to prove that certain relations hold for committed numbers, in particular that a committed number is the product of two other committed numbers.

In [8] efficient statistically zero-knowledge and computationally sound proof protocols are proposed for proving relations in modular arithmetic (addition, multiplication, exponentiation) between committed numbers. On common input $(C_a, C_b, C_c, C_v, par_{\text{COM}})$ the protocols are *proofs of knowledge* of (a, b, c, v) and $sk_{\text{COM}} := (sk^a_{\text{COM}}, sk^b_{\text{COM}}, sk^c_{\text{COM}}, sk^v_{\text{COM}})$ with:

$$\begin{aligned}
\textsc{Open}(C_a, par_{\text{COM}}, a, sk^a_{\text{COM}}) = (a, \top) \ \wedge \\
\textsc{Open}(C_b, par_{\text{COM}}, b, sk^b_{\text{COM}}) = (b, \top) \ \wedge \\
\textsc{Open}(C_c, par_{\text{COM}}, c, sk^c_{\text{COM}}) = (c, \top) \ \wedge \\
\textsc{Open}(C_v, par_{\text{COM}}, v, sk^v_{\text{COM}}) = (v, \top) \ \wedge \\
(a \ op \ b) \equiv c \bmod v
\end{aligned}$$

These protocols are statistical zero-knowledge and computationally sound under the discrete logarithm assumption for the underlying group. We denote them as

$$\text{POK}((C_a, C_b, C_c, C_v); (a, b, c, v), sk_{\text{COM}} : (a \ op \ b) \equiv c \bmod v),$$

with $op \in \{+, *, \exp\}$ and refer to [8] for the details of these protocols. As we do not need to prove modular relations, but only integer relations, we may fix C_v in the protocol as a commitment on a sufficiently large prime $v \in \mathcal{M}$, such that no overflow occurs or apply zero-knowledge proofs for integer arithmetic relations (see e.g. [15] for a zero-knowledge proof system for the multiplication relation). We will denote these protocols as

$$\text{POK}((C_a, C_b, C_c); (a, b, c), sk_{\text{COM}} : (a \ op \ b) = c),$$

Proving that a committed number is in an interval. Furthermore, we require an efficient zero-knowledge proof protocol for proving that a committed number is in an interval $[l, u]$. On common input (C_a, par_{COM}) the proof protocol is a *proof of knowledge* of (a, sk^a_{COM}) with: $\textsc{Open}(C_a, par_{\text{COM}}, a, sk^a_{\text{COM}}) = (a, \top) \wedge a \in [l, u]$. The protocol proposed in [7], applied to the commitments of [18, 15], is statistically zero-knowledge in the random oracle model and computationally sound. By setting the interval appropriately this zero-knowledge proof can be used to prove that a committed value is positive. A recent alternative to this range proof is due to Lipmaa [30]. This protocol uses Lagrange's four square decomposition of positive integer values to prove that a committed number is positive. We denote these protocols in short with $\text{POK}(C_a; (a, sk^a_{\text{COM}}) : a \geq 0)$.

3.2.2 Protocol

The protocol presented in this section depends on the detection statistic of the corresponding watermarking scheme. We show the protocol for a well-known blind watermarking scheme of Cox et al. [10]. However, the idea underlying this approach is general and adaptable to

other watermarking schemes and types of detection statistics, which can be computed using operators $+, *, -$. Blind detection[2] of a watermark $WM = (wm_1, \ldots, wm_k)$ in a stego-image $\overline{O}'$ works by computing the correlation value

$$corr = \frac{< \mathrm{DCT}(\overline{O}', k),\, WM >}{\sqrt{< \mathrm{DCT}(\overline{O}', k), \mathrm{DCT}(\overline{O}', k) >}} \tag{3.1}$$

between WM and the k largest DCT AC coefficients

$$\mathrm{DCT}(\overline{O}', k) = (\mathrm{DCT}(\overline{O}')_1, \ldots, \mathrm{DCT}(\overline{O}')_k).$$

Here, $< \cdot, \cdot >$ denotes the scalar product of two vectors. The value $corr$ is a measure of confidence for the presence of WM in $\overline{O}'$. The watermark is decided to be present in $\overline{O}'$ iff $corr \geq \delta$ holds for a predefined *detection threshold* δ. The detection threshold δ is a public parameter of the watermarking scheme, which determines the false-positive and false-negative probabilities.

The common inputs to the protocol are the committed watermark

$$C_{WM} = (C_{wm_1}, \ldots, C_{wm_k}),$$

the commitment parameter par_{COM}, and the stego-image $\overline{O}'$ in which the presence of the watermark should be proved. The quantity C_{wm_i} denotes the commitment to the watermark component wm_i. Additionally, $\mathcal{P}$ has the auxiliary input

$$sk_{\mathrm{COM}}^{WM} = (sk_{\mathrm{COM}}^{wm_1}, \ldots, sk_{\mathrm{COM}}^{wm_k}),$$

which is the secret opening information of C_{WM}. The tuple $(C_{WM}, sk_{\mathrm{COM}}^{WM})$ can be efficiently computed from WM using the commitment scheme.

In contrast to Cox et al., we assume that the watermark, DCT-coefficients and detection threshold are *integers* and not real numbers. Note, that this is no real constraint, because we can scale or quantize the real values appropriately. For efficiency reasons the following equivalent[3] detection criterion is used:

$$C := [(\underbrace{< \mathrm{DCT}(\overline{O}', k),\, WM >}_{A})^2 - \underbrace{< \mathrm{DCT}(\overline{O}', k), \mathrm{DCT}(\overline{O}', k) > * \delta^2}_{B}] \stackrel{?}{\geq} 0. \tag{3.2}$$

The message space of the commitment scheme must be large enough so that no values drop out when doing computations with the committed values. This can be done by choosing the parameters par_{COM} of the commitment scheme accordingly[4].

The protocol allowing $\mathcal{P}$ to prove to $\mathcal{V}$ that the watermark, hidden in commitments C_{WM}, is detectable in $\overline{O}'$ consists of the following steps:

[2] For a protocol allowing *non-blind* zero-knowledge watermark detection we refer to [5].

[3] Equivalency holds for $A \geq 0$, which is proven in step 4 of the detection protocol.

[4] Alternatively, we may choose smaller parameters and prove for each operation in zero-knowledge that no overflow occurred, e.g., using proofs from [7].

1. $\mathcal{P}$ and $\mathcal{V}$ compute $\mathrm{DCT}(\overline{O}', k)$.
2. $\mathcal{P}$ proves knowledge of the watermark components by performing the zero-knowledge sub-proofs $\mathrm{POK}(C_{wm_i}; (wm_i, sk_{\mathrm{COM}}^{wm_i}) : \textsc{Open}(C_{wm_i}, par_{\mathrm{COM}}, wm_i, sk_{\mathrm{COM}}^{wm_i}) = (wm_i, \top))$ for $i = 1, \ldots, k$.
3. $\mathcal{P}$ and $\mathcal{V}$ compute the commitment

$$C_A := \prod_{i=1}^{k} (C_{wm_i})^{\mathrm{DCT}(\overline{O}')_i}$$

   by exploiting the homomorphic property of the underlying commitment scheme.
4. $\mathcal{P}$ proves to $\mathcal{V}$ in zero-knowledge that C_A contains a value ≥ 0 by performing the sub-protocol $\mathrm{POK}(C_A; (A, sk_{\mathrm{COM}}^{A}) : A \geq 0)$.
5. $\mathcal{P}$ computes the value A^2, sends a commitment C_{A^2} to $\mathcal{V}$ and proves to $\mathcal{V}$ in zero-knowledge that C_{A^2} "contains" the square of the value contained in C_A by running the sub-protocol $\mathrm{POK}((C_A, C_A, C_{A^2}); (A, A, A^2), sk_{\mathrm{COM}} : (A * A) = A^2))$.[5]
6. $\mathcal{P}$ and $\mathcal{V}$ both locally compute the quantity B of the equivalent detection criterion C as given in Equation 3.2. Note that all necessary values are not concealed and publicly known.
7. Now, both $\mathcal{V}$ and $\mathcal{P}$ compute the commitment $C_C := C_{A^2} * (g^B)^{-1}$ on the value C.[6]
8. Finally, $\mathcal{P}$ proves to $\mathcal{V}$ in zero-knowledge that the value contained in C_C is ≥ 0. For this, $\mathcal{P}$ and $\mathcal{V}$ perform the sub-protocol $\mathrm{POK}(C_C; (C, sk_{\mathrm{COM}}^{C}) : C \geq 0)$. If $\mathcal{V}$ accepts this proof, it can be sure that the watermark hidden in C_{WM} is detectable in $\overline{O}'$ and it outputs $\top$.
9. If any of the local tests or zero-knowledge proofs fails the verifier considers the watermark as being not detectable and outputs $\perp$.

It can be shown (for details see [2]) that the above scheme is a computationally sound and statistically zero-knowledge watermark detection protocol in the random oracle model.

[5] Alternatively, we may use a sub-proof from [7] for proving that a committed number is a square.
[6] Note that g^B is a commitment on B with blinding factor 1.

Bibliography

[1] A. Adelsbach, S. Katzenbeisser, and A.-R. Sadeghi. Cryptography meets watermarking: Detecting watermarks with minimal or zero knowledge. In *European Signal Processing Conference (EUSIPCO 2002), Proceedings*, Toulouse (France), 2002.

[2] A. Adelsbach, S. Katzenbeisser, and A.-R. Sadeghi. Watermark detection with zero-knowledge disclosure. *ACM Multimedia Systems Journal*, 9(3):266–278, 2003.

[3] A. Adelsbach, M. Rohe, and A.-R. Sadeghi. Overcoming the obstacles of zero-knowledge watermark detection. In *Proceedings of ACM Multimedia Security Workshop*, pages 46–55, 2004.

[4] A. Adelsbach and A.-R. Sadeghi. Zero-knowledge watermark detection and proof of ownership. In *Proceedings of the Fourth International Workshop on Information Hiding*, volume 2137 of *Lecture Notes in Computer Science*, pages 273–188. Springer Verlag, 2001.

[5] A. Adelsbach and A.-R. Sadeghi. Zero-knowledge watermark detection and proof of ownership. In *Information Hiding*, volume 2137 of *Lecture Notes in Computer Science*, pages 273–288. Springer Verlag, 2001.

[6] M. Barni, F. Bartolini, and T. Furon. A general framework for robust watermarking security. *Signal Processing*, 82(10):2069–2084, 2003.

[7] F. Boudot. Efficient proofs that a committed number lies in an interval. In *Advancs in Cryptography—EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444. Springer Verlag, 2000.

[8] J. Camenisch and M. Michels. Proving in zero-knowledge that a number is the product of two safe primes. In *Advancs in Cryptography—EUROCRYPT 1999*, volume 1599 of *Lecture Notes in Computer Science*, pages 107–122. Springer Verlag, 1999.

[9] H. Choi, K. Lee, and T. Kim. Transformed-key asymmetric watermarking system. *IEEE Signal Processing Letters*, 11(2):251–255, February 2004.

[10] I. Cox, J. Kilian, T. Leighton, and T. Shamoon. A secure, robust watermark for multimedia. In *Information Hiding*, volume 1174 of *Lecture Notes in Computer Science*, pages 175–190. Springer Verlag, 1996.

[11] J. Cox, M. Miller, and J. Bloom. *Digital Watermarking*. Morgan Kaufmann, 2001.

[12] S. Craver. Zero knowledge watermark detection. In *Proceedings of the Third International Workshop on Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 101–116. Springer, 2000.

[13] S. Craver and S. Katzenbeisser. Copyright protection protocols based on asymmetric watermarking. In *Communications and Multimedia Security Issues of the New Century*, pages 159–170. Kluwer, 2001.

[14] S. Craver and S. Katzenbeisser. Security analysis of public-key watermarking schemes. In *Proceedings of the SPIE vol 4475, Mathematics of Data/Image Coding, Compression and Encryption IV with Applications*, pages 172–182, 2001.

[15] I. Damgård and E. Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In Yuliang Zheng, editor, *Advances in Cryptology—ASIACRYPT '2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 2002.

[16] P. Duhamel and T. Furon. An asymmetric public detection watermarking technique. In *Proceedings of the Third International Workshop on Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 89–100. Springer Verlag, 2000.

[17] J. J. Eggers, J. K. Su, and B. Girod. Public key watermarking by eigenvectors of linear transforms. In *Proceedings of the European Signal Processing Conference*, 2000.

[18] E. Fujisaki and E. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advancs in Cryptography—CRYPTO 1997*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer Verlag, 1997.

[19] T. Furon. *Use of watermarking techniques for copy protection.* PhD thesis, Ecole Nationale Supérieure des Télécommunications., 2002.

[20] T. Furon and P. Duhamel. An asymmetric watermarking method. *IEEE Trans. on Signal Processing*, 51(4):981–995, April 2003. Special Issue on Signal Processing for Data Hiding in Digital Media and Secure Content Delivery.

[21] T. Furon, I. Venturini, and P. Duhamel. An unified approach of asymmetric watermarking schemes. In P.W. Wong and E. Delp, editors, *Security and Watermarking of Multimedia Contents III*, San Jose, Cal., USA, January 2001. SPIE.

[22] O. Goldreich. *Foundations of Cryptography*, volume 1, Basic Tools. Cambridge University Press, 2001.

[23] O. Goldreich and Y. Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, 1994.

[24] K. Gopalakrishnan, N. Memon, and P. Vora. Protocols for watermark verification. In *Multimedia and Security, Workshop at ACM Multimedia*, pages 91–94, 1999.

[25] F. Hartung and B. Girod. Fast public-key watermarking of compressed video. In *International Conference on Image Processing (ICIP'97)*, volume I, pages 528–531, 1997.

[26] F. Hartung and B. Girod. Watermarking of uncompressed and compressed video. *Signal Processing*, 66(3):283–301, 1998.

[27] N. J. Hurley and G. C. M. Silvestre. Nth-order audio watermarking. In P. W. Wong and E. J. Delp, editors, *Security and Watermarking of Multimedia Contents IV, Proc. SPIE Vol. 4675*, pages 102–109, San Jose, CA, USA, 2002.

[28] T. Kalker. A security risk for publicly available watermark detectors. In *Proc. Benelux Inform. Theory Symp.*, Veldhoven, The Netherlands, May 1998.

[29] T. Y. Kim, T. Kim, H. Choi, K. Lee, and T. Kim. An asymmetric watermarking system with many embedding watermarks corresponding to one detection watermark. *IEEE Signal Processing Letters*, 11(3):375–378, March 2004.

[30] H. Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In C.S. Laih, editor, *Advances in Cryptology—ASIACRYPT '2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 398–415. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 2003.

[31] M. F. Mansour and A. H. Tewfik. Secure detection of public watermarks with fractal decision boundary. In *Proc. XI Europ. Signal Processing Conf., EUSIPCO'02*, Toulouse, France, 2002.

[32] M. L. Miller. Is asymmetric watermarking necessary or sufficient? In *Proc. XI Europ. Signal Processing Conf., EUSIPCO'02*, pages 291–294, Toulouse, France, 2002.

[33] J. Quisquater, L. Guillou, and T. Berenson. How to explain zero-knowledge protocols to your children. In *Advances in Cryptography—CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 628–631. Springer Verlag, 1989.

[34] A. De Rosa, M. Barni, F. Bartolini, V. Cappellini, and A. Piva. Optimum decoding of non-additive full frame dft watermarks. In *Proc. of Third International Workshop on Information Hiding*, pages 160–172, Dresden, Germany, January 1999. Springer LNCS 1768.

[35] J. Smith and C. Dodge. Developments in steganography. In *Proc. of Third International Workshop on Information Hiding*, pages 77–87, Dresden, Germany, January 1999. Springer LNCS 1768.

[36] R. G. van Schyndel, A. Z. Tirkel, and I. D. Svalbe. Key independent watermark detection. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 580–585, 1999.