DWT Collusion Resistant Video Watermarking Using Tardos Family Codes

Abdul Rehman
IRT b-com
Cesson-Sevigné, France
abdul.rehman@b-com.com

Gaëtan Le Guelvouit
IRT b-com
Cesson-Sevigné, France
gaetan.leguelvouit@b-com.com

Jean Dion
IRT b-com
Cesson-Sevigné, France
jean.dion@b-com.com

Frédéric Guilloud
IMT Atlantique, Lab-STICC
Brest, France
frederic.guilloud@imt-atlantique.fr

Matthieu Arzel
IMT Atlantique, Lab-STICC
Brest, France
matthieu.arzel@imt-atlantique.fr

***Abstract*—A fingerprinting process is an efficient means of protecting multimedia content and preventing illegal distribution. The goal is to find individuals who were engaged in the production and illicit distribution of a multimedia product. We investigated discrete wavelet transform (DWT) based blind video watermarking strategy tied with probabilistic fingerprinting codes to avoid collusion among higher-resolution videos. We used FFmpeg to run a variety of collusion attacks (e.g., averaging, darkening, and lighten) on high resolution video and compared the most often suggested code generator and decoders in the literature to find at least one colluder within the necessary code length. The Laarhoven codes generator and nearest neighbor search (NNS) decoder outperforms all other suggested generators and decoders in the literature in terms of computational time, colluder detection and resources.**

***Index Terms*—Collusion, Video watermarking, Fingerprinting codes**

I. Introduction

The digital revolution and peer-to-peer networks have significantly influenced our daily lives, particularly digital information piracy [9]. Consumers may now access digital content and services at any time and from any location. In this context, copyright crimes such as free distribution, illicit usage, and unauthorized sharing of copyrighted digital content are getting more common. Videos are perhaps the most vulnerable multimedia content [1], and unauthorized individuals are spreading videos for their own gain and profit. As a result, digital operations may suffer and the business model may be impacted.

The problem of super re-distribution can be addressed using fingerprint-based approaches [17, 5, 19]. Each consumer's fingerprint (a unique watermark, or customer ID) is embedded in the video clip. One of the design characteristics of fingerprinting scheme is that it generates a code for each client that allows the distributor to identify the authentic users. If a digital pirate chooses to publish his fingerprinted content publicly, the content owner can get a copy, extract the fingerprint, connect it to the responsible user, and take necessary action. To avoid being recognized, digital pirates may work together to make a mixed copy of the information [6], denoted as a collusion. To avoid the collusion of media content, collusion-resistant fingerprinting approaches are needed like Tardos codes [20], which provide an acceptable trade-off between code length and decoding time. Tardos codes properties have been improved in order to make them more resistant to collusion attempts [3, 10]. Previous fingerprinting systems attempted to strike a balance between tracing code and watermarking technology in order to provide a tracing method that could survive a variety of collusion attacks [2, 26, 23, 6]. Nonetheless, they only utilized a normal average collusion attack with excessively lengthy fingerprinting codes to evaluate the durability of these systems. There are various joint fingerprinting and encryption (JFE) techniques [25, 18, 21] that employ DWT watermarking, which is resistant to brute force, differential, and statistical attacks. However, there is no justification for the various components of fingerprint schemes (e.g. multimedia compression, collusion resistance codes, error probabilities and collusion attacks). Other works demonstrate the usage of fingerprinting code with DWT watermarking [13, 12], but the resilience has only been proven with an average attack on gray-scale images.

In this paper, we investigate the embedding of probabilistic fingerprinting codes using DWT watermarking technique. We first compare the combination of several probabilistic fingerprinting codes generation with several decoders without embedding. We then embed these probabilistic fingerprinting codes using DWT into high resolution color videos using FFmpeg. The performance are evaluated for several attacks (darken, lighten and average) and we show that taking into account the

embedding do change the conclusion regarding the best code generation - decoder combination.

Our paper is organized as follows. In Section II we remind the basic techniques to generate and decode fingerprinting codes related to probabilistic fingerprinting codes. Then in Section III, we describe the two modes considered in this paper: the first mode considers collusion attacks without embedding whereas the second one considers collusion attacks with DWT-based embedding. Then the performance and comparison of these techniques are presented. Based on these comparisons, a discussion is proposed in Section IV and a conclusion is provided in Section V.

II. Probabilistic Code Schemes

The probabilistic fingerprinting code schemes are divided into two steps: (1) generation, which involves creating a code or identifier for each authentic users, and (2) decoding, which involves finding the users that are participating in the collusion.

A. Generation

Boneh and Shaw [4] proposed a binary method alphabet size ($q = 2$) that combined a partially randomized inner code with a deterministic outer code. The code-length for the scheme is $m = {c_0}^4 \log \frac{n}{\eta} \log \frac{1}{\eta}$, where η is the chance of a False Positive (FP) error, n is total number of users and c_0 is maximum number of colluders. It also gave a lower bound on the expected code-length $m = \Omega(c_0 \log \frac{1}{c_0 \eta})$. Tardos [20] showed an even tighter bound of $m = \Omega({c_0}^2 \log \frac{1}{\varepsilon_1})$, where ε_1, represents the likelihood that one specific innocent user is accused, and he gave a fully randomized binary code with $m = 100{c_0}^2 ln \frac{1}{\varepsilon_1}$, that achieves that bound. Binary Tardos codes are generated as follows. The content owner creates a $n \times m$ binary matrix $\mathbf{X}$ where each row $\mathbf{X_{j:}}$ corresponds to a codeword (or identifier) for user j as shown in Fig. 1. Each entry X_{ji} in column i (with $i = 1, ..., n$) follows

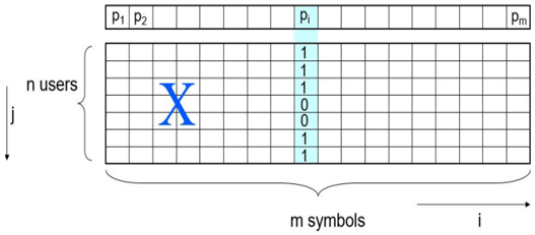

Fig. 1. Code matrix $\mathbf{X}$ created by content owner

a Bernoulli distribution with parameter p_i. Parameters p_i are drawn from an arcsine distribution [20] and are denoted random biases. For Tardos codes, the random biases generation is detailed in Appendix A Algorithm 1. The q-ary generalization [3] makes use of a Dirichlet distribution and, as stated, at $q = 2$, it reduces to the arcsine distribution. In [10] Nuida proposed a discrete distribution for p_i that is based on c_0. If the actual collusion size $c \leq c_0$, the modified discrete distribution enhances decoding; however, when the collusion size $c > c_0$ the modified discrete distribution has lower performance. Finally, Laarhoven and de Weger in [22] proved that Nuida discrete distribution [10] asymptotically converges to the arcsine distribution of Tardos codes. For Laarhoven codes, the bias distribution is detailed in Appendix A Algorithm 2.

In this article, we chose to use and compare both the Tardos codes with their continuous distribution and the Laarhoven codes with their discrete distribution.

B. Decoding

Let $\mathbf{y}$ denote the extracted identification bit vector corresponding to a collusion attack, and thus not to a legitimate user identifier. In this context, decoding means to retrieve the legitimate users who participated to the generation of the illegitimate copy. To this aim, scores are calculated using some scoring function g: $s_{ji} = g(X_{ji}, y_i, p_i)$. User j is declared as pirate if the cumulative score $\sum_{i=1}^{m} s_{ji} > Z$ for a given threshold Z. Another strategy consist in accusing the users with the greatest cumulative scores.

Along with Tardos codes, the optimal scoring function without embedding, regardless of the collusion attack, denoted as Tardos-Skoric scoring function, was proposed in [20], and then generalized in [3, 24]. The Tardos-Skoric function is provided in [3] by Equation (1).

$$g(X_{ji}, y_i, p_i) := \begin{cases} +\sqrt{\frac{1-p_i}{p_i}^{2y_i-1}(p_i)^{1-2y_i}} & \text{if } X_{ji} = y_i; \\ -\sqrt{\frac{p_i}{1-p_i}^{2y_i-1}(1-p_i)^{1-2y_i}} & \text{if } X_{ji} \neq y_i; \end{cases} \tag{1}$$

When the maximum number of colluders c_{max} can be defined and assuming apriori information about the collusion attack represented by parameter θ_c, an optimal decoder is provided in [16], and will be denoted *Desoubeaux* decoder hereafter. The Desoubeaux scoring function is reminded in [16] by Equation (2).

$$g_{\theta_c}(X_{ji}, y_i, p_i) := \log\left(\frac{\mathbb{P}(Y=y_i|X_{ji},p_i,\theta_c)}{\mathbb{P}(Y=y_i|p_i,\theta_c)}\right) \tag{2}$$

Here θ_c is collusion model and the probabilities can be found in [7, Eq. (8-9)]. The decoder presented in [14] is still based on the knowledge of c_{max} but without any need for apriori information on the collusion attack. We will refer to this score as the *Laaroven* score. It is described in [14] by Equation (3).

$$g(X_{ji}, y_i, p_i) := \begin{cases} \log\left(1 + \frac{1}{c_{max}}\left(\frac{1-p_i}{p_i}\right)^{2y_i-1}\right) & \text{if} \quad X_{ji} = y_i; \\ \log\left(1 - \frac{1}{c_{max}}\right) & \text{if} \quad X_{ji} \neq y_i; \end{cases} \tag{3}$$

Finally, the last decoder considered in this article will be denoted as NNS and was proposed in [15] as a nearest

neighbourhood search. The NNS score does not require any a priori on c_{max} nor on the collusion attack and is detailed in [15] by Equation (4).

$$g(X_{ji}, y_i, p_i) := \frac{(2X_{ji}-1)(2y_i-1)}{\sqrt{p_i(1-p_i)}} \quad (4)$$

The decoding time and computing resources are crucial criteria to consider when determining a decoder's efficiency. The complexity for all the above described decoders are given in Table I, where $\rho \leq 1$ is determined by the initial settings of the fingerprinting scheme [15], and $k \leq c_{max}$ is the length needed in the generalized linear decoders [16]. From Table I, we can forsee that the NNS score decoder is the less time-consuming one.

TABLE I
THIS TABLE DISPLAYS THE DECODING TIME NEEDED BY ALL THE DECODER IN TERMS OF THEIR NEED TO ACCUSE THE COLLUDERS.

Decoders	Decoding complexity
Tardos-skoric	$O(mn)$
Laarhoven	$O(mnc_{max})$
Desoubeaux	$O(mn^k c_{max}\theta_c)$
NNS	$O(mn^\rho)$

III. Proposed Scheme and Experiments

In order to compare the performance of the code generator and decoders, we considered two modes to analyze collusion codes and their decoding scores, namely mode A and mode B. In mode A, we consider collusion codes without embedding. In mode B, we consider a DWT and FFmpeg based embedding of the collusion codes in 4K resolution videos. In the following, the studied collusions are described according to the considered mode.

A. Mode A: no embedding

A collusion of c colluders uses their identifiers $\boldsymbol{X}_j$, $j \in \mathcal{J}$, where $\mathcal{J}$ is the set of c colluders, to create a pirated copy $\mathbf{y}$. Note that we assume that if all of the c colluders have the same bit at position i in their codes, then the y_i will be given this bit also. The attacks are then defined when at least one bit is different among the users. Hereafter, we define in this context 2 attacks: majority and minority, as illustrated in Table II for $c = 3$ colluders.

To compare the performance of the code generation schemes and the decoder scores, we conducted 100 Monte Carlo simulations with a fixed $\mathbf{X}$ matrix, code-length $m = 128$ bits and $c = 12$ colluders that are randomly selected from a range of users.

The average number of detected colluders is provided in Table III when using the Tardos code generation and in Table IV when using the Laarhoven code generation. We can observe that the Desoubeaux and Laarhoven scores have the best detection rate whatever the generation technique. Also, when comparing the 2 generation techniques, the best performance is obtained with the Laarhoven code generator.

TABLE II
COMMON COLLUSION ATTACKS WITHOUT EMBEDDING FOR $c = 3$ COLLUDERS

Attack	collusion codes
Majority	1 0 0 0 1 1 0 0 0 1 0 1
	1 1 0 0
Minority	1 0 0 0 1 1 0 0 0 1 0 1
	0 0 0 1

TABLE III
MODE A: AVERAGE DETECTED COLLUDERS USING DIFFERENT DECODERS FOR TARDOS CODES WITH $c = 12$

Attack	n	Tardos score	Laarhoven score	Desoubeaux score	NNS score
Majority	30	3.95	6.86	6.65	5.81
	50	3.85	4.87	4.96	3.95
	100	1.91	1.96	2.86	2.91
minority	30	3.96	6.62	6.56	5.69
	50	3.83	4.53	4.48	3.64
	100	1.71	1.78	2.85	1.79

TABLE IV
MODE A: AVERAGE DETECTED COLLUDERS USING DIFFERENT DECODERS FOR LAARHOVEN CODES WITH $c = 12$

Attack	n	Tardos score	Laarhoven score	Desoubeaux score	NNS score
Majority	30	4.81	6.62	6.96	5.92
	50	4.62	4.79	4.81	4.98
	100	1.89	1.81	2.82	2.89
minority	30	5.77	5.32	7.96	5.83
	50	3.96	4.96	6.51	3.62
	100	2.00	1.98	2.98	2.99

B. Mode B: Embedding with DWT watermarking

Since their publication, Tardos codes have been employed into several type of embeddings. In [26], they were used to fingerprint H.264/AVC utilizing a wide spectrum resilient watermarking approach. Huge limitations however on the memory usage and massive code length prevent them to be considered for real time implementations. Tardos fingerprinting algorithm and zero-bit broken arrows watermarking approach for photos were proposed in [23]. They've demonstrated that this combination has ruled out fusion attacks. In [2], a unique collusion-secure Tardos code based fingerprinting strategy for 3D movies was presented, which uses a conventional least significant bit (LSB) replacement for all frames of both the 2D video and the depth map components. However, here again extremely long codes were employed without considering real-time attacks on

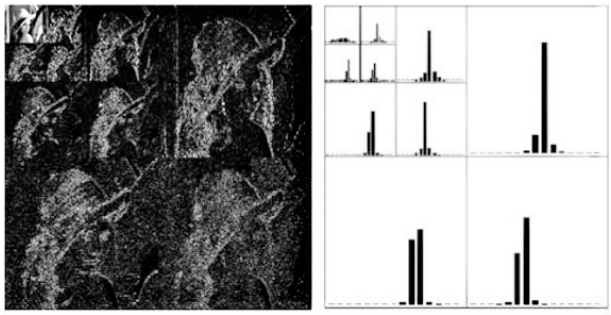

Fig. 2. Result of 3-level DWT decomposition of Lena using CDF 9/7 and its coefficient distribution in different sub-bands (Ref. [11].)

the videos. The embedding of Tardos code with DWT watermarking was demonstrated in [13, 12], but the resilience has only been proven with an average attack on gray-scale images.

In this article, we propose to embed the collusion codes using the DWT watermarking technique and FFmpeg blending filters on high resolution (4K) videos. We analyzed watermarked image using the Cohen-Daubechies-Feauveau (CDF9/7) wavelets because according to [8], CDF9/7 wavelet outperforms all the other approaches of DWT watermarking. Fig. 2 depicts a three-level split of Lena as well as the dispersion of its coefficients into several sub-bands. There are 10 sub-bands. So far, many watermarking approaches that take use of the wavelet transform's multi-resolution capacity to embed a watermark in the host image have shown that a 3-level wavelet decomposition is enough to successfully obstruct the watermark (see e.g. [11]). So we chose to examine our proposed strategy also with a 3-level wavelet decomposition. The embedding procedure is illustrated in the top of Fig. 3 and consist in the following steps:

- Create a random image of size 360 x 640.
- In this image, merge fingerprinting codes.

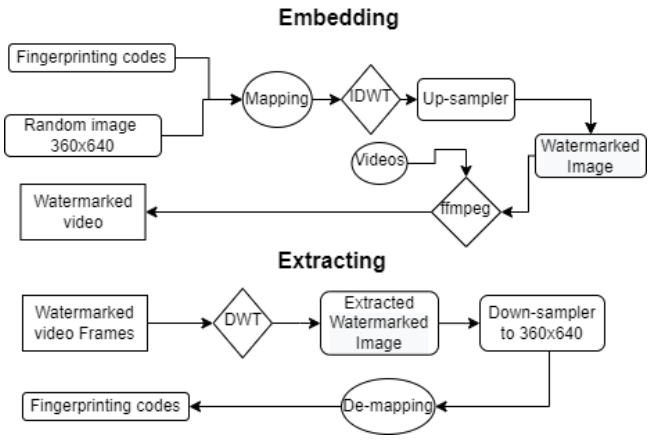

Fig. 3. DWT-based Embedding and analyzing of fingerprinting codes in videos

- Apply a 3-level inverse discrete wavelet transform (IDWT) to the given image.
- Up-sample the image to the resolution of the video frame.
- Encode the watermarked image into video using the FFmpeg blend filter.

The collusion code extraction procedure starts with the DWT of the watermarked video frames as illustrated in the lower part of Fig. 3.

Three different collusion attacks (average, darken, and lighten filters) are simulated using FFmpeg blending filters as shown in Table V. The number of detected colluders against the collusions described previously for the different decoding scores is given in Table VI for Tardos codes and in Table VII for Laarhoven codes for $c = 12$ colluders and a codelength $m = 128$ bits. We observe that we can detect more than one colluders using Desoubeaux and NNS decoders for average and darken attacks for $n = 100$ users. On the other hand, when the attack is lighten, we can detect only one colluder as demonstrated in Table VI. In contrast, with Laarhoven codes we may find multiple colluders for all attacks utilizing Desoubeaux's, laarhoven and NNS decoders as shown in Table VII.

TABLE V
FFMPEG BLENDING FILTERS FOR COLLUSION ATTACKS (MODE B)

FFmepg function	Mathematical method	Attacks
Lighten	$\max(A, B)$	Majority
Darken	$\min(A, B)$	Minority
Average	$\frac{A+B}{n}$	Average

TABLE VI
MODE B: DETECTED COLLUDERS USING DIFFERENT DECODERS FOR TARDOS CODES WITH $c = 12$

Filter	n	Tardos score	Laarhoven score	Desoubeaux score	NNS score
Darken	30	3	5	5	6
	50	3	3	3	4
	100	1	1	2	2
Lighten	30	3	4	4	5
	50	3	4	3	4
	100	1	1	1	1
Average	30	5	6	6	5
	50	2	2	2	3
	100	1	1	2	2

IV. DISCUSSION

1) About the choice of the Bias distribution: In this article, 2 different bias distribution functions were studied: discrete (Laarhoven) and continuous (Tardos) distribution. Based on our simultations, we determined that the Laarhoven discrete bias distribution function performs slightly better than the continuous Tardos one, as shown in Table VI and VII due to the shorter code length constraint.

TABLE VII
MODE B: DETECTED COLLUDERS USING DIFFERENT DECODERS FOR LAARHOVEN CODES WITH $c = 12$

Filter	n	Tardos score	Laarhoven score	Desoubeaux score	NNS score
Darken	30	3	6	5	5
	50	4	4	4	4
	100	1	2	2	2
Lighten	30	5	5	7	5
	50	3	4	4	4
	100	1	1	2	2
Average	30	4	6	7	5
	50	5	6	6	5
	100	1	2	2	2

2) About the code length: The fingerprinting code length is totally determined by the fingerprinting technique's generating parameters, error probability (ε_1), and the number of colluders c. However, it should be as low as possible. We used 128-bit fingerprinting code lengths for our research, because our aim was to find at least one colluder. However, if the aim is to accuse all the colluders while accommodating a larger number of users, the code length should be raised according to design parameters of fingerprinting scheme.

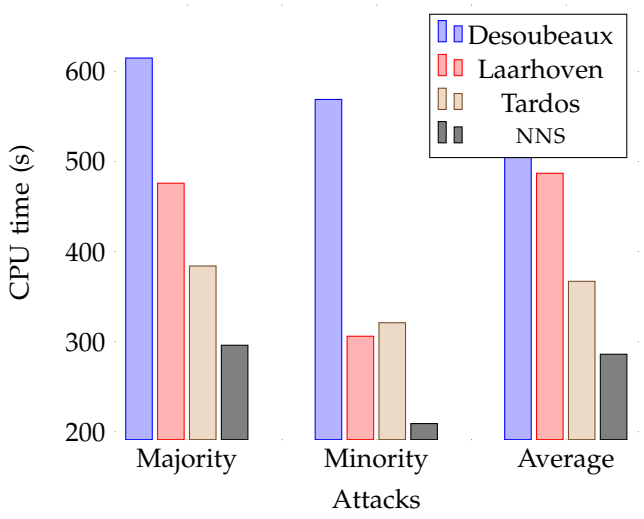

Fig. 4. Tracing time for all the decoders after embedding in videos (Mode B) for three attacks with parameters $n = 100$, $c = 12$ averaged over 100 *trials*

3) About the choice of the decoder score: Without embedding (Mode A), Desoubeaux outperforms all other decoders and still depends on the collusion attack model θ_c, which is not so convenient in realistic conditions. When embedding (Mode B) is taken into account, the NNS decoder surpasses all the others, in terms of decoding performance as shown in Fig.4, complexity, as it takes 51% less CPU time compare to the most recommended Desoubeaux decoder for all the proposed attacks. In this context and considering realistic embedding, we advocate utilizing the NNS decoder.

V. CONCLUSION

In this paper, we investigated a DWT video watermarking system based on the Tardos and Laarhoven collusion codes on high resolution videos, and their performance against realistic attacks. To this aim, decoding strategies based on four different scores were studied, both in term of performance to detect colluders and in term of complexity: Tardo-Skoric, Laarhoven, Desoubeaux and NNS scores. Simulations show that using Laarhoven codes decoded by the NNS decoder offer the best performance with the lowest complexity, which makes this tandem a good candidate for real-time implementation on the protection of high resolution video watermarking against collusion attacks. As only a part of the colluders can be found, a perspective of the work is to improve the capability of the decoder by performing an iterative processing.

APPENDIX A
ALGORITHMS

In this appendix, the generation algorithm of the random biases related to the Tardos codes (resp. Laarhoven codes) is described in Algorithm 1 (resp. Algorithm 2).

Algorithm 1 Random bias for Tardos codes

1: let $t = \frac{1}{300c}$, where c is number of colluders and let $t' = arcsin\sqrt{t}$.
2: $\forall i \in [1, m]$; draw a random r_i according to uniform distribution in $[t', \frac{\pi}{2-t'}]$.
3: $\forall i \in [1, m]$; calculate $p_i = sin^2(r_i)$.

Algorithm 2 Random bias for Laarhoven codes

1: $\forall i \in [1, m]$; draw r_i according to uniform distribution from $\left(\frac{3\pi}{(8c+4)}, \frac{7\pi}{(8c+4)} .... \frac{\pi}{2}, -\frac{3\pi}{(8c+4)}\right)$, where c is number of colluders.
2: $\forall i \in [1, m]$; calculate $p_i = sin^2(r_i)$.

REFERENCES

[1] Karama Abdelhedi, Faten Chaabane, and Chokri Ben Amar. "A SVM-Based Zero-Watermarking Technique for 3D Videos Traitor Tracing". In: *Advanced Concepts for Intelligent Vision Systems*. Ed. by Jacques Blanc-Talon et al. Cham: Springer International Publishing, 2020, pp. 373–383. ISBN: 978-3-030-40605-9.

[2] Karama Abdelhedi et al. "Toward a Novel LSB-based Collusion-Secure Fingerprinting Schema for 3D Video". In: trans. by Nicolas Tsapatsoulis et al. Computer Analysis of Images and Patterns. Cham: Springer International Publishing, 2021, pp. 58–68. ISBN: 978-3-030-89128-2.

[3] M. Celik B. ÅåkoriÄĞ S. Katzenbeisser. "Symmetric Tardos fingerprinting codes for arbitrary alphabet sizes". In: 46.2 (2008), pp. 137–166. URL: https://eprint.iacr.org/2007/041.pdf.

[4] J. Boneh D. and Shaw. "Collusion-secure fingerprinting for digital data". In: 44.5 (1998), pp. 1897–1905.

[5] Teddy Furon. "Traitor Tracing". In: *Multimedia Security 1: Authentication and Data Hiding* (2022), pp. 189–218.

[6] Teddy Furon and Mathieu Desoubeaux. "Tardos codes for real". In: *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. 2014, pp. 24–29. DOI: 10.1109/WIFS.2014.7084298.

[7] Teddy Furon and Luis Pérez-Freire. "Worst case attacks against binary probabilistic traitor tracing codes". In: *CoRR* abs/0903.3480 (2009). arXiv: 0903.3480. URL: http://arxiv.org/abs/0903.3480.

[8] Nicholas Hopper, David Molnar, and David Wagner. "From Weak to Strong Watermarking." In: *IACR Cryptology ePrint Archive* 2006 (Jan. 2006), p. 430.

[9] Yun Hu et al. "Towards a privacy protection-capable noise fingerprinting for numerically aggregated data". In: *Computers & Security* 119 (2022), p. 102755. ISSN: 0167-4048. DOI: 10.1016/j.cose.2022.102755. URL: https://www.sciencedirect.com/science/article/pii/S016740482200150X.

[10] H. Watanabe and H. Ima K. Nuida M. Hagiwara. "An improvement of discrete Tardos fingerprinting codes". In: 52.3 (2007), pp. 339–362. ISSN: 0925-1022. DOI: 10.1007/s10623-009-9285-z. URL: http://dx.doi.org/10.1007/s10623-009-9285-z.

[11] P. Kumsawat, K. Attakitmongcol, and A. Srikaew. "A new approach for optimization in image watermarking by using genetic algorithms". In: *IEEE Transactions on Signal Processing* 53.12 (2005), pp. 4707–4719. DOI: 10.1109/TSP.2005.859323.

[12] Minoru Kuribayashi. "Simplified MAP Detector for Binary Fingerprinting Code Embedded by Spread Spectrum Watermarking Scheme". In: *IEEE Transactions on Information Forensics and Security* 9.4 (2014), pp. 610–623. DOI: 10.1109/TIFS.2014.2305799.

[13] Minoru Kuribayashi and Hans Georg Schaathun. "Image fingerprinting system based on collusion secure code and watermarking method". In: *2015 IEEE International Conference on Image Processing (ICIP)*. 2015, pp. 2120–2124. DOI: 10.1109/ICIP.2015.7351175.

[14] Laarhoven. "Capacities and Capacity-Achieving Decoders for Various Fingerprinting Games". In: *In ACM Workshop on Information Hiding and Multimedia Security* (2014). URL: arXiv:1401.5688v1[cs.IT]22Jan2014.

[15] Thijs Laarhoven. "Nearest neighbor decoding for Tardos fingerprinting codes". eng. In: 2019. URL: arXiv:1902.06196v1[cs.CR]17Feb2019.

[16] W. Puech and G. Le Guelvouit. M. Desoubeaux C. Herzet. "Enhanced Blind Decoding of Tardos Codes with New Map-Based Functions." In: *IEEE 15th International Workshop on Multimedia Signal Processing (MMSP),* (Oct. 2013). Pula, Italie, URL: https://arxiv.org/pdf/1305.7038.pdf.

[17] David Megias, Minoru Kuribayashi, and Amna Qureshi. "Survey on Decentralized Fingerprinting Solutions: Copyright Protection through Piracy Tracing". In: *Computers* 9.2 (2020). ISSN: 2073-431X. DOI: 10.3390/computers9020026. URL: https://www.mdpi.com/2073-431X/9/2/26.

[18] Reham Mostafa, Alaa el-din Riad, and Hamdy El-Minir. "A Novel JFD Scheme for DRM Systems Based on DWT and Collusion Resistance Fingerprint Encoding". In: *International Arab Journal of Information Technology* 13 (Jan. 2016), p. 842.

[19] Amna Qureshi and David Megias. "Blockchain-based P2P multimedia content distribution using collusion-resistant fingerprinting". In: *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2019, pp. 1606–1615. DOI: 10.1109/APSIPAASC47483.2019.9023054.

[20] Gábor Tardos. "Optimal Probabilistic Fingerprint CodesâĹŮ". In: 2003, pp. 116–125. URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.8911&rep=rep1&type=pdf.

[21] Alok Tripathi, Rajiv Pandey, and Amarjeet Singh. "Simulating Tardos Finger Printing Codes under Randomized Bits Collusion Attacks". In: *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*. 2021, pp. 201–205. DOI: 10.1109/ICCCIS51004.2021.9397094.

[22] T. Laarhoven and B. de Weger. "Discrete distributions in the tardos scheme, revisited". In: (2018). URL: arXiv:1302.1741v2[cs.CR]29Apr2013.

[23] Fuchun Xie, Teddy Furon, and Caroline Fontaine. "On-Off Keying Modulation and Tardos Fingerprinting". In: *Proc. ACM Multimedia and Security* (Jan. 2008). DOI: 10.1145/1411328.1411347.

[24] Tatsuya Yasui et al. "Near-Optimal Detection for Binary Tardos Code by Estimating Collusion Strategy". In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 2069–2080. DOI: 10.1109/TIFS.2019.2956587.

[25] Conghuan Ye et al. "Joint fingerprinting and encryption in hybrid domains for multimedia sharing in social networks". In: *Journal of Visual Languages & Computing* 25.6 (2014), pp. 658–666. ISSN: 1045-926X. DOI: 10.1016/j.jvlc.2014.10.020. URL: https://www.sciencedirect.com/science/article/pii/S1045926X1400113X.

[26] William Puech Zafar Shahid Marc Chaumont. "H.264/AVC video watermarking for active fingerprinting based on Tardos code". eng. In: *Signal, Image and Video Processing, Springer Verlag* 7.4 (2013). DOI: 10.1007/s11760-013-0483-9ff. URL: https://hal-lirmm.ccsd.cnrs.fr/lirmm-00807061.