On Multicast Fingerprinting and Collusion Security [*]

Josep Domingo-Ferrer, Francesc Sebé and Antoni Martínez-Ballesté
Rovira i Virgili University of Tarragona
Dept. of Computer Engineering and Maths
Av. Països Catalans 26
E-43007 Tarragona, Catalonia
e-mail {josep.domingo,francesc.sebe,antoni.martinez}@urv.net

Abstract

Intellectual property protection of multimedia is a thorny and largely unsolved problem. Despite its practical limitations, copy detection is an excellent complement to copy prevention. Fingerprinting techniques make redistributor tracing possible by relying on invisible watermarks embedded in the content sold which are unique for each particular buyer. That uniqueness of fingerprints is not easy to achieve in multicast transmission. We argue in this paper that encryption-based solutions are the most realistic option to fingerprint multicast transmission. However, the few existing encryption-based methods in the literature are vulnerable to buyer collusions. We propose here an encryption-based algorithm which provides collusion security at the cost of transmitting the content twice.

Keywords: *Copyright protection, Copy detection, Multicast, Watermarking, Fingerprinting.*

1 Introduction

As pointed out in [8], intellectual property rights (IPR) protection is a tricky issue. Most copy prevention schemes have been broken by hackers after some time, the most prominent and recent failure being the one of the DVD copy prevention system [5]. Thus, a promising complement to copy prevention is copy detection. If copy prevention is defeated but there is some copy detection mechanism left, neutralizing the latter is not painless for the attacker, as it normally entails some penalty in terms of content quality.

In copy detection, an imperceptible mark is embedded into the content being sold. The techniques for imperceptible embedding of marks into content are known as *watermarking*. *Fingerprinting* [17] is a copy detection technique in which the embedded mark, now called fingerprint, identifies the buyer of the content. In this way, upon finding an illegal copy of a content, recovery of the embedded fingerprint allows identification of the buyer who had her copy redistributed.

A good watermarking system must be *robust* at least against accidental —signal processing— attacks, that is, the embedded watermark should survive those attacks. Ideally, watermarking should also survive intentional attacks using knowledge of the watermarking algorithm being used (*tamper-proofness*). In practice, tamper-proofness is not achieved by current watermarking methods, so a common option is trying to keep secret the watermarking algorithm being used (sometimes with the help of some trusted hardware). Indeed, Kerckhoff's assumption that keys should be the only secrets is harder to meet in watermarking than in cryptography: the main goal of an attacker is not to "decrypt" the embedded watermark, but rather to destroy it in order to be able to take full advantage of the resulting "unwatermarked" content. Even without knowledge of the secret embedding key, an attacker knowing the principles of the embedding algorithm is often able to distort the content so that the watermark is no longer recoverable.

When watermarks are used to embed fingerprints unique for each buyer, an additional class of attacks becomes feasible. A set of buyers can collude and analyze the differences between the fingerprinted copies each of them has obtained; the goal of this analysis is to locate and remove/modify as many bits of their fingerprints as possible in view of obtaining a copy whose embedded fingerprint no longer identifies any of the colluders.

[*]Work partly funded by the Spanish Ministry of Science and Education under project SEG2004-04352-C04-01 "PROPRIETAS". This work was performed in part while the first author was a Visiting Fellow at Princeton University.

If the algorithm to construct the fingerprints is independent from the watermarking algorithm used to embed them, then the so-called marking assumption [3] can be made: the fingerprinting algorithm needs only guard against collusion attacks, because the rest of single-user attacks are taken care of by the underlying watermarking algorithm.

Several techniques for building collusion-resistant fingerprints for *unicast* contents have been proposed in the literature [3, 6, 7, 14, 12, 15, 16]. In those unicast techniques, the source is assumed to embed a different fingerprint in the content sent to each buyer. The literature on *multicast* fingerprinting is less plentiful, let alone on collusion-resistant multicast fingerprinting. The requirement that each multicast receiver (buyer) should get a distinctly and uniquely fingerprinted content may even seem contrary to the very idea of multicast (bandwidth saving by sending the same content to all receivers).

In multicast transmission, the same content is sent by the source to all receivers. Thus, the concept of fingerprinting, implying a different version for each receiver, seems to contradict this kind of transmission. At a closer look, though, there is no contradiction. Even if the multicast source transmits a single version of the content, the multicast receivers can get distinct versions provided that a diversification process is performed by the intermediate routers along the multicast tree or by the receivers themselves. Ways to enforce content diversification can be based on trusted receiver devices or just on encryption.

Therefore, solutions for multicast transmission fall into one of three approaches:

- Distributed marking fingerprinting
- Trusted receiver devices
- Encryption-based fingerprinting

We will justify that practical solutions for multicast fingerprinting are encryption-based. However, it will be shown that encryption-based methods in the literature are vulnerable to buyer collusions and sometimes wasteful in terms of bandwidth.

Section 2 reviews multicast fingerprinting based on distributed marking. Section 3 analyzes multicast fingerprinting based on trusted devices. Section 4 deals with encryption-based multicast fingerprinting. In Section 5, a collusion-resistant multicast fingerprinting algorithm is described. Section 6 contains some conclusions and lines for future research.

2 Distributed marking fingerprinting

Under this approach, intermediate routers along the path from the multicast source to a receiver are in charge of diversifying the content the receiver gets in such a way that it is distinct from the content delivered to other receivers. Two systems in this line are next discussed.

The Watercasting system [4] is based on dividing the content into packets and preparing several marked versions of the same packet. All packet versions are sent to the multicast tree and routers must discard some of the versions, so that receivers at the leaves of the multicast tree will get the complete content, but marked differently depending on which version they received for each packet. This system has weak points:

- the policy of packet discarding at each router must be centrally scheduled by the source;
- assuming that a trusted router infrastructure is available may not be realistic (*e.g.* routers on the Internet are not trusted);
- if d marked versions of each packet are prepared, then the source needs to transmit d times the actual content (bandwidth waste).

The WHIM[1] system [9] is another example of distributed marking fingerprinting, where the content is marked by distributed trusted multicast routers that embed a different portion of the mark into the content as the latter is being transmitted along the routing path. In this way, recovering the fingerprint from an illegal copy allows tracing the path followed by the content, which discloses its legal recipient (who is the likely redistributor). Despite of this being an attractive solution, it has the weak point that deploying a publicly-available trusted distributed system is not an easy task. Furthermore, in [9] the problem of collusion attacks is not addressed.

3 Trusted receiver devices

In [2], a multicast source sends encrypted content to receivers, who can decrypt it using a tamper-resistant device (*e.g.* smart card). Each receiver device is identified by its serial number sn and stores a private key SK_{sn}; there is a public key PK_{sn} corresponding to SK_{sn}. Broadcast content is encrypted under a symmetric session key; this session key is sent encrypted under the public keys of the receivers who should be

[1]Watermarking multicast video with a Hierarchy of InterMediaries.

able to access the content; when receiver device sn decrypts a session key using SK_{sn}, it also fingerprints the decrypted content by embedding sn in it.

The strong point of this solution is that it can be readily implemented. Its weak points are that:

- collusions are not resisted unless codewords of a collusion-secure fingerprinting scheme are embedded in lieu of serial numbers sn;

- second but not least, if trusted receiver devices could really be trusted, copy detection could be abandoned in favor of copy prevention (but copy prevention based on trusted devices has mostly been cracked, *e.g.*, CD copy prevention, DVD copy prevention, encrypted TV).

4 Encryption-based fingerprinting

As mentioned above, under this approach the multicast source encrypts the content before transmission and the differences between the decryption keys given to the receivers cause each receiver to recover a uniquely marked content after decryption. Intermediate routers in the multicast tree are not required to play any role in this process.

Since our own proposal will be encryption-based, we discuss in some detail the encryption-based systems in the literature.

In [1], a system called Chameleon was proposed as the first encryption-based solution for multicast fingerprinting. Chameleon combines encryption and fingerprinting in a computationally efficient way. The idea is that encryption is performed on the plaintext at the source using a group key K_S to produce the ciphertext that is multicast to all n receivers. Slightly different decryption keys from the set $K_R = \{K_1, K_2, \cdots, K_n\}$ are distributed to the users such that, when decryption is performed on the ciphertext, the result is slightly different for each user.

The authors of [1] propose a specific implementation for fingerprinting multicast uncompressed audio streams. This implementation adapts a block cipher for encryption in output feedback mode to ensure that only the least significant bits (LSBs) of the plaintext audio are changed when the content is decrypted with a user's decryption key. This guarantees that the fingerprint will not cause perceptual degradation.

The main strong point of Chameleon is its computational efficiency. Its main weak points are that:

- Chameleon mixes fingerprinting with watermarking, so that the marking assumption discussed in Section 1 does not apply and the security of fingerprinting cannot be analyzed separately from the security of watermarking. Since the watermark is an LSB one, noise addition or content compression/decompression may badly damage the fingerprint even in the absence of collusions.

- In spite of content being transmitted just once, bandwidth usage is not that efficient, because preserving LSB watermarks precludes compression.

- The decryption keys to be distributed to receivers are very large, which is not attractive in multicast scenarios.

- As acknowledged by its authors, Chameleon fingerprinting can hardly survive collusion attacks by five or more receivers, who can use a bit-wise majority voting to erase the fingerprint from the LSBs and come up with a plaintext that cannot be traced.

In [13], an encryption-based multicast fingerprinting system is proposed which separates mark embedding from encryption at the source. This solution is based on dividing the multimedia stream into packets and can be described as follows:

- The source divides the media stream into k packets and generates $2k$ random encryption keys.

- For each packet P_i, the source generates two different watermarked versions P_i^0 and P_i^1.

- P_i^0 is encrypted by the source under a random key k_i^0 and P_i^1 is encrypted under a random key k_i^1. Both encrypted versions are multicast to all receivers.

- For each packet P_i, each receiver is given one of the two decryption keys k_i^0, k_i^1. Thus, if a receiver is assigned a sequence of keys $\{k_1^0, k_2^1, k_3^1, k_4^0, \cdots\}$, she will be able to recover $P_1^0, P_2^1, P_3^1, P_4^0, \cdots$ after decryption.

Thus, each receiver obtains a slightly different version of the overall content by composition of the packet versions she is able to decrypt.

After finding an unlawfully redistributed copy of the content, the sequence of keys assigned to the subscriber who was the last legal receiver can be determined from the packet versions forming the copy. The embedded fingerprint is recovered from the sequence of keys as follows: the fingerprint i-th bit is 0 if the subscriber had key k_i^0 and is 1 if the subscriber had key k_i^1.

The strong point of this scheme is that fingerprint construction (*i.e.* the algorithm for assigning decryption keys to receivers) is clearly separated from the underlying watermarking algorithm used to create the two versions of each packet. Thus, assuming that a sufficiently robust watermarking algorithm is used, by the marking assumption the fingerprinting algorithm needs only deal with collusion attacks. However, there are also weak points:

- In the basic form of the scheme, the bandwidth usage is doubled over that of normal multicast, because two encrypted versions of each packet are multicast. Parviainen and Parnes suggest that bandwidth usage might be reduced by only doubling (and thus watermarking) one in every x packets. However, they admit that, if this is done, a malicious attacker becomes aware that all non-doubled packets can be redistributed without any possible tracing.
- The scheme remains collusion-resistant only if the number of colluders is reduced *and* the collusion-generated stream consists of a random composition of packets. We showed in [11] that assuming a random collusion strategy is unrealistic and that three attackers using a so-called minority strategy can always defeat the traitor tracing properties of the Parviainen-Parnes scheme.

4.2.1 A collusion attack to Parvianen-Parnes

We formalize here the attack sketched in [11]. As mentioned above, the algorithm [13] provides collusion security as long as the number of colluders remains reduced and the collusion-generated stream consists of a random composition of packets. In case of collusion, the receiver whose fingerprint is nearest (using Hamming distance) to the recovered one will be accused.

However, colluders may use non-random strategies. One of those is the so-called *minority strategy*, discussed in what follows. Let us assume that k-bit long fingerprints unique for each receiver are embedded into the content. Now, a collusion of only three receivers (3-collusion) can defeat the tracing properties of the system [13] as follows:

Attack 1

1. *The colluders first detect fragments of the stream for which they have received different versions. In fragments detectable by a 3-collusion, there are always two colluders who have the same version, which is called the* majority *fragment; in contrast, the third colluder has a different version, called the* minority *fragment.*
2. *Colluders generate a new stream where they always use the* minority *fragment in case of a detectable fragment.*

Although not much is said on [13] about how decryption keys are assigned to receivers, at one point it is suggested that, in order to avoid the source having to store all key assignments of receivers, the k-bit masks specifying which keys are assigned to each receiver can be generated using a cryptographically secure (pseudo)random number generator. This results in (pseudo)random fingerprints. The lemma and the theorem below assume that random fingerprints are used.

Lemma 1 *If participants in a 3-collusion were assigned random k-bit fingerprints in the scheme [13], one would expect each colluder's stream to contain $k/4$ undetectable fragments (identical for the three colluders), $k/2$ majority fragments and $k/4$ minority fragments.*

Proof: The result is straightforward if one realizes that the i-th bit in a receiver's fingerprint indicates which version of the i-th stream packet the receiver gets. $\square$

Theorem 1 *If participants in a 3-collusion against the scheme [13] use a minority strategy, none of them can be traced.*

Proof: After a collusion using a minority strategy, by Lemma 1, each colluder's stream is expected to differ in its $k/2$ majority fragments from the collusion-generated stream. Thus, the Hamming distance between the collusion-generated fingerprint and each colluder's fingerprint is expected to be $k/2$. But $k/2$ is the average Hamming distance between any two k-bit fingerprints; in particular between the collusion-generated fingerprint and any innocent receiver's fingerprint. Therefore, colluders cannot be traced. $\square$

A new architecture called JFD and inspired on Chameleon has been recently proposed for multicast fingerprinting [10].

The source extracts the perceptually relevant features from the multimedia content X and selectively encrypts them with key K_S. Then the source multicasts the encrypted content Y to n receivers. Each receiver has previously been given a decryption key K_i by the source. The decryption key set denoted by $K_R = \{K_1, K_2, \cdots, K_n\}$ is designed jointly with the source key K_S so that an imperceptible and indelible fingerprint is embedded in the content after decryption. Thus, the JFD architecture is basically equivalent to Chameleon with the addition that the source only encrypts features selected as perceptually relevant.

The authors of [10] give a single example of a JFD scheme applied to video fingerprinting. The raw video frame is encoded through a discrete cosine transform (DCT) and quantized, like in the JPEG compression algorithm [18]. However, unlike in JPEG, the DCT is taken on the entire image, rather than on 8x8 pixel blocks. From the quantized DCT coefficients thus obtained, a set of coefficients in the low and midfrequency region are identified that are perceptually significant. This set of coefficients is partitioned into n subsets. These subsets are all sign-scrambled (*i.e.*, the sign is arbitrarily flipped depending on the key) during encryption and some are left scrambled during decryption for the purpose of forming the fingerprint (similarly to what was done in Chameleon).

There are two types of encryption keys: the first serve as pointers to subsets of coefficients to encrypt, and the second as the scrambling key that dictates the order in which the coefficients are permuted within a given subset. The receiver is given a unique subset of keys K_i for decrypting only p of the n encrypted subsets. The remaining $k = n - p$ subsets are hidden from the receiver and their unique sign bit signature along with their concealed positions in the frame constitutes the fingerprint.

As is common in watermarking, there is a triangular tradeoff between imperceptibility, robustness and capacity:

- *Imperceptibility vs robustness.* Let M_j be the number of coefficients in the j-th hidden subset. As M_j increases, the diversity of the embedded fingerprint increases, which increasing the probability of its correct retrieval (robustness). However, a large M_j means that a large number of coefficients are left scrambled, which reduces perceptual quality (imperceptibility).
- *Imperceptibility vs capacity.* If the number k of hidden subsets is reduced, imperceptibility increases, but the fingerprint length (capacity) decreases.

When analyzing the JFD scheme suggested in [10], it turns out that it is has similar strong and weak points as Chameleon:

- On the strong side, computational efficiency is similar to Chameleon: DCT computation certainly causes some overhead, but encryption/protection is only needed for perceptually significant DCT coefficients, because the non-significant coefficients can be left unencrypted (unprotected). Regarding bandwidth efficiency, the JFD scheme allows for compression (unlike Chameleon) and requires content to be transmitted only once (unlike Parviainen-Parnes). Thus, bandwidth efficiency is higher than in the preceding schemes.
- On the weak side, we have:
  - Like Chameleon, the proposed JFD scheme still mixes fingerprinting with watermarking. Thus the security of fingerprinting cannot be analyzed separately from the security of watermarking. The robustness results reported in [10] against single-user signal-processing attacks are not comprehensive: they are limited to JPEG compression attacks and are not particularly encouraging.
  - Just like Chameleon, the proposed JFD scheme is not designed to be collusion-resistant.
  - Watermarking by randomly flipping the sign of DCT coefficients is known to cause considerable degradation in terms of perceptual quality [19]. Note in the JFD scheme forming the fingerprint requires a substantial number (literally some subsets) of DCT coefficients to be left sign scrambled after decryption.

5 A collusion-resistant multicast fingerprinting algorithm

When attempting to obtain a collusion-resistant fingerprinting algorithm, it seems a natural strategy to take one of the above encryption-based systems as a starting point.

In Chameleon, the embedding process is governed by a pseudo-random keystream generator. Thus, imposing some structure on the embedded marks to make them collusion-resistant (*e.g.* taking the marks as codewords of a collusion-secure code such as [3]) seems hardly possible without completely changing the design of Chameleon. A similar remark applies to the JFD scheme.

Furthermore, even if in the case of Chameleon one managed to give a collusion-secure structure to the marks, the lack of robustness of the LSB watermarking used causes the marks to be erasable by simple signal-processing attacks. The algorithm [13] performs worse in terms of bandwidth but might be more easily rendered collusion-resistant.

The vulnerability of the [13] scheme to collusions can be repaired by using a decryption key assignment based on Boneh-Shaw collusion-secure fingerprinting codes [3]. Boneh and Shaw built binary codes secure against collusions of up to c colluders out of a group of n buyers (c-secure codes). Parameters c and n are chosen together with a security parameter $\epsilon > 0$ which is the maximum acceptable probability of failure in colluder re-identification. Codewords have length

$$l = 32c^4 \log_2(2n/\epsilon) \log_2(8cL/\epsilon) \qquad (1)$$

where $L = 2c \log_2(2n/\epsilon)$. Our collusion-resistant multicast fingerprinting algorithm is given next.

Algorithm 1

1. *Given the maximum number n of potential receivers in a multicast session, the source chooses parameters c (maximum tolerable collusion size) and ϵ.*
2. *A c-secure code with n codewords and probability ϵ of re-identification failure is constructed. The codewords have length l (see Equation (1)).*
3. *The source divides the multimedia stream into l packets and generates two watermarked encrypted versions of each packet.*
4. *The source assigns to each receiver a different codeword of the c-secure code, so that the receiver gets the decryption keys specified by that codeword as follows: for the i-th packet, if the i-th bit of the assigned codeword is 0, the receiver gets k_i^0; otherwise, she gets k_i^1.*

When a redistributed copy is found, the fingerprint is recovered in the same manner described in Section 4.2. After recovery, the tracing algorithm of [3] is applied, which identifies one of the colluders with probability at least $1 - \epsilon$, as long as the collusion size is not greater than c.

6 Conclusions and future research

We have surveyed and analyzed the literature on multicast content fingerprinting. The conclusion of that analysis is that encryption-based methods are the most realistic option, but all previous methods in this class are vulnerable to collusion attacks and some of them are wasteful in terms of bandwidth (uncompressed or double content transmission required).

We have proposed a modification of a previous encryption-based scheme to obtain a collusion-resistant multicast fingerprinting scheme. Its only remaining drawback is that it requires transmitting the content twice. Future research should be directed to improving bandwidth usage.

References

[1] R. J. Anderson and C. Manifavas, "Chameleon–A new kind of stream cipher", in *Fast Software Encryption-FSE'97*, LNCS 1267, pp. 107-113, 1997.

[2] F. Bao, "Multimedia content protection by cryptography and watermarking in tamper-resistant hardware", in *Proceedings of the 2000 ACM Workshops on Multimedia*, pp. 139-142, 2000.

[3] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data", *IEEE Transactions on Information Theory*, vol. 44, pp. 1897-1905, Sep. 1998.

[4] I. Brown, C. Perkins and J. Crowcroft, "Watercasting: distributed watermarking of multicast media", in *Proceedings of the First International Workshop on Networked Group Communication*, pp. 286-300, 1999.

[5] DeCSS, `http://www.lemuria.org/DeCSS`

[6] J. Dittman, A. Behr, M. Stabenau, P. Schmitt, J. Schwenk and J. Ueberberg, "Combining digital watermarks and collusion secure fingerprints for digital images", in *Proc. SPIE Conference on Electronic Imaging'99, Security and Watermarking of Multimedia Contents*, vol. 41, pp. 171-182, 1999.

[7] J. Domingo-Ferrer and J. Herrera-Joancomartí, "Short collusion-secure fingerprints based on dual binary Hamming codes", *Electronics Letters*, vol. 36, no. 20, pp. 1697-1699, Sep. 2000.

[8] S. Haber, B. Horne, J. Pato, T. Sander and R. E. Tarjan, "If piracy is the problem, is DRM the answer?", in E. Becker et al. (eds.) *Digital Rights Management-DRM'03*, LNCS 2770, pp. 224-233, 2003.

[9] P. Judge and M. Ammar, "WHIM: Watermarking multicast video with a hierarchy of intermediaries", in *Proc. of NOSSDAV 2000*, Chapel Hill, NC, Jun. 2000.

[10] D. Kundur and K. Karthik, “Video fingerprinting and encryption principles for digital rights management”, *Proceedings of the IEEE*, vol. 92, no. 6, pp. 918-932, Jun. 2004.

[11] A. Martínez-Ballesté, J. Domingo-Ferrer and F. Sebé, “Fingerprinting schemes for multicast delivery”, in *International Conference on Information Technology: Research and Education-ITRE’2003*, Newark NJ, USA, Aug. 11-13, 2003.

[12] D. Naor and M. Naor, “Protecting cryptographic keys: the trace-and-revoke approach”, *IEEE Computer*, vol. 38, no. 7, pp. 47-53, July 2003.

[13] R. Parviainen and P. Parnes, “Large scale distributed watermarking of multicast media through encryption”, in *Proc. of IFIP Communications and Multimedia Security 2001*, Norwell MA: Kluwer Academic Publishers, pp. 149-158, 2001.

[14] R. Safavi-Naini and Y. Wang, “Collusion-secure q-ary fingerprinting for perceptual content”, in T. Sander (ed.) *Digital Rights Management-DRM’01*, LNCS 2320, pp. 57-75, 2002.

[15] F. Sebé and J. Domingo-Ferrer, “Collusion-secure and cost-effective detection of unlawful multimedia redistribution”, *IEEE Transactions on Systems, Man and Cybernetics-C*, vol. 33, no. 3, pp. 382-389, 2003.

[16] W. Trappe, M. Wu, J. Z. Wang and K. J. R. Liu, “Anti-collusion fingerprinting for multimedia”, *IEEE Transactions on Signal Processing*, vol. 51, no. 3, pp. 1069-1087, 2003.

[17] N. Wagner, “Fingerprinting”, in *Proc. of the 1983 IEEE Symposium on Security and Privacy*, Oakland, pp. 18-22, 1983.

[18] G. Wallace, “The JPEG still picture compression standard”, *Communications of the ACM*, vol. 34, no. 4, pp. 30-44, 1991.

[19] W. Zeng and S. Lei, “Efficient frequency domain selective scrambling of digital video”, *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 118-129, Mar. 2003.