ARES project
CONSOLIDER-INGENIO 2010
CSD2007-00004
Workpackage 3 - Task 5 (WP3.T5)
Practical anonymous fingerprinting schemes
Deliverable Report

Josep Domingo-Ferrer*, David Megías†,

*Universitat Rovira i Virgili,
UNESCO Chair in Data Privacy,
Department of Computer Engineering and Mathematics,
Av. Països Catalans 26, E-43007 Tarragona, Catalonia,
e-mail josep.domingo@urv.cat

†Universitat Oberta de Catalunya,
Estudis d'Informàtica, Multimèdia i Telecomunicació,
Internet Interdisciplinary Institute (IN3),
Rambla del Poblenou, 156, 08018 Barcelona, Catalonia, Spain
e-mail dmegias@uoc.edu

October 23, 2012

Contents

Abstract

This report summarizes the advances carried out in anonymous fingerprinting within the ARES project. Two different solutions for anonymous fingerprinting are presented in this report. The first one is based on a game theoretic approach with multicast distribution of contents, whereby buyers cooperate by engaging in a protocol to produce subsequent copies of the fingerprinted contents. The second approach is based on a pure peer-to-peer distribution scenario where the recombination of "segments" of the fingerprints of "seed" buyers produces unique identifiers for subsequent customers. In the latter case, cooperation of buyers is not needed for embedding, but for traitor tracing. Both systems satisfy the requirements and objectives specified by the project objectives, namely, privacy preservation for honest buyers and scalability of the proposed solutions for a practical application scenario.

Chapter 1

Introduction

Copyright protection techniques have gained widespread attention by both academia and industry in the recent years. Home Internet access and the increased bandwidth of communications have contributed to the explosion of copyright-breaking copying of digital contents. In this context, fingerprinting emerged as a convenient technology to fight against unlawful digital content distribution [9, 7].

Fingerprinting techniques consist of embedding a transparent watermark into the protected content in such a way that a unique identifier exists for each buyer of the content. This identifier can be extracted later on and might be used to trace and match an illegal redistributor of the content. This makes it possible to undertake the appropriate legal actions against such treacherous buyers. Fingerprinting schemes can be classified in three different categories [10], namely symmetric, asymmetric and anonymous. In symmetric fingerprinting, the embedding of the fingerprint is performed by the merchant only and, thus, it provides no valid evidence of a treacherous behavior of a buyer (since the merchant herself could be the illegal redistributor). In asymmetric fingerprinting, the embedding is performed using a protocol designed in such a way that only the buyer obtains the fingerprinted copy of the content. This makes it possible to prove the illegal redistributor's treachery to a third party. Finally, anonymous fingerprinting retains the asymmetric property and also protects the privacy of buyers, whose identity is only revealed and disclosed in case of illegal redistribution.

From the point of view of a buyer, anonymity is a valuable property and several protocols have been proposed for anonymous fingerprinting. However, current anonymous fingerprinting proposals in the literature place a substantial computational and communication burden on the merchant. The merchant's overhead is a relevant issue, since it will possibly result in buyer anonymity not being offered or offered at higher price by the merchant so that the latter can still enjoy some profit margin. Hence, the possibility of reducing the merchant's burden and the flexibility of choosing the watermarking technology freely among the best state-of-the-art techniques are worth investigating. This report focuses on proposing two multicast approaches to the anonymous fingerprinting problem which meet these two goals and shows proofs of concept with practical implementations of the proposed systems. The idea is to transfer the burden of a centralized fingerprinting technology to a distributed network of buyers who

will collaborate to produce further copies of the fingerprinted contents.

If a content is to be distributed to a group of N receivers, one option is for the content sender to engage in N unicast transmissions, one for each intended receiver, and another option is a single multicast transmission to the entire group. The multicast option certainly has the advantage of being quicker and more bandwidth-efficient from the sender's point of view. However, the unicast approach has the strong point of allowing the sender to fingerprint the content sent to each receiver by embedding a different serial number in each sent copy, in view of detecting and tracing unlawful redistribution of the content. Note that the multicast approach does not allow fingerprinting, as all receivers get exactly the same content. Hence, the unicast approach, in spite of its inefficiency, seems more suitable when the sender is a merchant selling content and the receivers are buyers.

Peer-to-peer (P2P) distribution of content appears as a third option blending some of the advantages of the unicast and multicast solutions. P2P distribution of all types of files and contents has become extremely popular with the increased bandwidth of home Internet access in the last few years. BitTorrent [1], Kademlia [31] or eDonkey2000 [2] are widely known examples of P2P file sharing protocols. In addition, P2P file sharing applications are not restricted to this use, and some companies are also exploiting the P2P distribution paradigm as a way of saving server bandwidth and speeding up the downloads of their products (such as multimedia contents and software updates, *e.g.* [4]). Indeed, when using a P2P network for content distribution, the merchant only needs to establish direct connections with one or a few seed buyers, say $M \ll N$ buyers, and send them copies. The content is further spread over the P2P network by those seed buyers. The challenge is how to ensure that the P2P spread content is still traceable in case of redistribution.

The type of fingerprinting relevant to this deliverable report is anonymous fingerprinting. In anonymous fingerprint schemes, the merchant does not have access to the identities or the fingerprints of buyers, which protects their security and privacy. Initial anonymous fingerprinting proposals depended on unspecified multiparty secure computation protocols [37, 17]. In [18], an anonymous fingerprinting protocol completely specified from the computational point of view and based on committed oblivious transfers was described. In [21], anonymous fingerprinting protocols were simplified under the assumption that a tamper-proof smart card was available on the buyer's side.

Many anonymous fingerprinting schemes exploit some homomorphic property of public-key cryptography [27, 40, 29, 34, 39, 38]. These schemes allow embedding the fingerprint in the encrypted domain (using the public key of the buyer) in such a way that only the buyer obtains the decrypted fingerprinted content. However, developing a practical system using these ideas appears difficult, because public-key encryption expands data and substantially increases the communication bandwidth required for transfers [26].

In [10], a different approach using group signatures was suggested, but it requires bit commitment and a zero-knowledge proof, implying a large overhead and high communicational costs. In the proposal of [8], the system's efficiency is enhanced due to the suppression of zero-knowledge proofs and public-key cryptography is not required in the embedding scheme. However, a secure two-party computation protocol is used

between the merchant and each buyer to transfer the fingerprinted content. In [26], any secure watermarking scheme (for which no proof of existence is available) may be used to develop an anonymous fingerprinting protocol if the watermark embedder provides a certain level of security. Although the proposed approach avoids the costs of homomorphic cryptography, a practical application of that idea is not presented. Another proposal to reduce the burden of anonymous fingerprinting on the buyer's side is presented in [11], where powerful servers would perform the most costly parts of the protocols. In any case, all the proposed anonymous fingerprinting systems incur high computational and communicational burdens at the buyer's and/or at the merchant's side, due to the use of some highly demanding technology (public-key encryption of the contents, secure multiparty protocols or zero-knowledge proofs, among others). Some of them also require specific embedding schemes which are not among the most robust or secure ones, or a secure watermarking system that is not proven to exist.

In this report, we propose two novel solutions to overcome these drawbacks, since the use of public-key cryptography is restricted to the transmission of short bit strings and is not applied to the multimedia content itself. In addition, the proposed schemes decentralize the transmission of the content using a network of peer buyers, thereby reducing the bandwidth needed by the merchant. The proposed protocols have been submitted in revised form [22, **?**] to ISI JCR-indexed journals and are currently under second review.

The rest of this report is organized as follows. Chapter 2 presents a multicast approach for fingerprinting based on a game theoretical approach and the rational cooperation between buyers (players). The scheme is shown to be implementable using an existing watermarking system. Chapter 3 presents a completely different approach, with a purely P2P distribution scenario and recombined automatic fingerprints which are generated as contents are downloaded by buyers from other peers. Both solutions offer practical implementable protocols overcoming many of the drawbacks of the methods published in the literature. Finally, Chapter 4 summarises the conclusions of this report.

Chapter 2

Distributed Multicast of Fingerprinted Content Based On a Rational Peer-to-Peer Community

2.1 Contribution of this chapter

We specify a protocol [22] whereby a sender manages to distribute a digital content to an unlimited number of receivers in such a way that:

- The content carries a different anonymous fingerprint for each receiver, so that unlawful content redistribution can be tracked; honest receivers stay anonymous.
- The sender does not need to fingerprint and send the content individually to each receiver; one fingerprinting and one unicast transmission by the server to one collaborative receiver are enough to bootstrap the process.
- Receivers are rationally interested to collaborate in forwarding and fingerprinting the content to other interested receivers (we call such rational collaboration co-utility); thanks to anonymous fingerprinting, intermediate receivers do not know the identities of the receivers they are forwarding the fingerprinted content to.

Section 2.2 gives some background on game theory, co-utility and anonymous fingerprinting. Section 2.3 describes the protocol and justifies its security. Section 2.4 argues the rational involvement by peers in game-theoretic terms and shows that our protocol achieves co-utility. Section 2.5 contains experimental results of a proof of concept. Section 2.6 summarizes conclusions and future research issues.

2.2 Background

2.2.1 Basics of game theory

A game is a protocol between a set of N *players*, $\{P^1, \cdots, P^N\}$. Each player P^i has her own *set of possible strategies*, say S_i. To play the game, each player P^i selects a strategy $s_i \in S_i$. We use $s = (s_1, \cdots, s_N)$ to denote the vector of strategies selected by the players and $S = \Pi_i S_i$ to denote the set of all possible ways in which players can pick strategies.

The vector of strategies $s \in S$ selected by the players determines the outcome for each player, which can be a payoff or a cost. In general, the outcome will be different for different players. To specify the game, we need to give, for each player, a preference ordering on these outcomes by giving a complete, transitive, reflexive binary relation on the set of all strategy vectors S. The simplest way to assign preferences is by assigning, for each player, a value for each outcome representing the payoff of the outcome (a negative payoff can be used to represent a cost). A function whereby player P^i assigns a payoff to each outcome is called a utility function and is denoted by $u_i : S \longrightarrow \mathbb{R}$.

For a strategy vector $s \in S$, we use s_i to denote the strategy chosen by player P^i and s_{-i} to denote the $(N-1)$-dimensional vector of the strategies played by all other players. With this notation, the utility $u_i(s)$ can also be expressed as $u_i(s_i, s_{-i})$.

A strategy vector $s \in S$ is a *dominant strategy solution* if, for each player P^i and each alternate strategy vector $s' \in S$, it holds that

$$u_i(s_i, s'_{-i}) \geq u_i(s'_i, s'_{-i}) \tag{2.1}$$

In plain words, a dominant strategy s is the best strategy for each player P^i, independently of the strategies played by all other players.

A strategy vector $s \in S$ is said to be a *Nash equilibrium* if, for all players P^i and each alternate strategy $s'_i \in S_i$, it holds that

$$u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i})$$

In plain words, no player P^i can change her chosen strategy from s_i to s'_i and thereby improve her payoff, assuming that all other players stick to the strategies they have chosen in s. A Nash equilibrium is self-enforcing in the sense that once the players are playing such a solution, it is in every player's best interest to stick to her strategy. Clearly, a dominant strategy solution is a Nash equilibrium. Moreover, if the solution is strictly dominant (*i.e.* when the inequality in Expression (2.1) is strict), it is also the unique Nash equilibrium. See [35] for further background on game theory.

2.2.2 Co-utility

We recall here the co-utility paradigm, which we introduced under the name general coprivacy in [19, 20]. The following definition is simpler but equivalent to the one used in our previous papers.

Definition 1 (Co-utility) *Let* Π *be a game with self-interested, rational players* P^1, $\cdots$, P^N *with* $N > 1$. *Game* Π *is said to be* co-utile *with respect to the vector* $U = (u_1, \cdots, u_N)$ *of utility functions if there exist at least two players* P^i *and* P^j*, having strategies* s^i *and* s^j*, respectively, such that: i)* s^i *involves* P^i *expecting co-operation from* P^j*; ii)* s^j *involves* P^j *co-operating with* P^i*; iii)* (s^i, s^j) *is an equilibrium for* P^i *and* P^j *in terms of* u_i *and* u_j*, respectively. In other words, there is co-utility between* P^i *and* P^j*, for some* $1 \leq i, j \leq N$ *with* $i \neq j$*, if the best strategy for* P^i *involves expecting co-operation from* P^j *and the best strategy for* P^j *is to co-operate.*

If the equilibrium in Definition 1 is a Nash equilibrium, we have *Nash co-utility*. If the utility functions U in Definition 1 only consider privacy, co-utility becomes the plain coprivacy notion introduced in [19, 20]; if utilities only consider security, we could speak of co-security; if they only consider functionality, co-utility becomes co-functionality.

2.2.3 Anonymous fingerprinting

Let $D_0 \in \{0,1\}^*$ denote some digital content (bit-string) some of whose bits can be changed in such a way that (1) the result remains "close" to D_0 (where "close" means "with a similar utility"), but (2) without knowing which particular bits were changed, altering a "good portion" of these bits is impossible without rendering the content useless. The changed bits are usually called a mark or watermark; if bits are changed differently for each user receiving the content, the mark can also be called fingerprint. The algorithm used to embed a mark while satisfying the previous two conditions is called a watermarking algorithm; to embed a fingerprint can also be termed "to fingerprint". The second requirement above is actually the marking assumption stated in [9].

As mentioned in the introduction above, the type of fingerprinting relevant to our report is anonymous fingerprinting. All the systems referred to in the introduction (Chapter 1) share a common drawback: the computational and communicational burdens for the merchant are quite high, due to the use of at least one of the following highly demanding technologies: public-key cryptography, bit commitment schemes, zero-knowledge proofs, secure multiparty computation schemes or secret sharing. In addition, some of them can only work with watermarking technologies which are not among the most robust and secure ones or even rely in some watermarking system for which no proof of existence has been provided yet.

In this chapter, we seek to mitigate the above performance shortcomings. Our protocols can be used with any of the above anonymous fingerprinting schemes. We borrow from [10] the following generic model of an anonymous fingerprinting protocol.

Definition 2 *An anonymous fingerprinting scheme involves a merchant, a buyer and a registration center. Let* c *denote the maximal size of a collusion of buyers against which the scheme is secure. An anonymous fingerprinting scheme consists of the following five procedures.*

FKG-RC: *A probabilistic key setup algorithm for the registration center. Its outputs are the center's secret key x_C and its public key y_C, which is published in an authenticated manner.*

FReg: *A probabilistic two-party protocol (*FReg-RC, FReg-B*) between the registration center and the buyer. Their common input is the buyer's identity ID_B and the center's public key y_C. The center's secret input is its secret key x_C. The buyer's output consists of some secret x_B and related information y_B. The center obtains and stores y_B and ID_B.*

FPrint: *A two-party protocol* (FPrint-M, FPrint-B) *between the merchant and the buyer. Their common input consists of y_C. The merchant's secret input is D_0 and a transaction number j and her output is a transaction record t_j. The buyer's secret input is x_B and y_B, and her output consists of a copy $D_B \in \mathcal{D}$, where $\mathcal{D}$ is the set of all close copies of D_0.*

FRec: *This may be a protocol or an algorithm whose purpose is to recover the identity of a/the fraudulent buyer responsible for the redistribution of a version $\tilde{D} \in \mathcal{D}$:*

- *It is a two-party protocol between the merchant and the registration center if the merchant needs the help of the registration center. The merchant's input is a copy $\tilde{D} \in \mathcal{D}$, all transaction records t_i and perhaps the original content D_0. The center's input consists of its secret key x_C and its list of y_B's and ID_B's. The merchant's output is a/the fraudulent buyer's identity together with a proof p that this buyer indeed bought a copy of D_0, or $\perp$ in case of failure (*e.g.*, if more than c buyers colluded to produce $\tilde{D}$).*
- *It is an algorithm run by the merchant alone if the merchant can determine a/the fraudulent's buyer identity with just $\tilde{D}$, all transaction records t_i and perhaps the original content D_0.*

Whether the original content D_0 is needed for identity recovery depends on the underlying watermarking method used to embed the fingerprint in the content: a watermarking method is said to allow blind detection if only the marked content $\tilde{D}$ and the embedded mark contained in the transaction record are needed for recovery; methods which need also D_0 are called informed watermarking. In return for their smaller flexibility, informed methods tend to be more robust to content manipulation; see Chapter 2 of [15] for a more detailed discussion.

FVer: *A verification algorithm, that takes as input the identity ID_B of an accused buyer, the public key y_C of the registration center, and a proof p, and outputs 1 iff the proof is valid.*

The solution in [10] guarantees the following properties:

Correctness: All protocols terminate successfully whenever players are honest (no matter how other players behaved in other protocols).

Anonymity and unlinkability: Without obtaining a particular D_B , the merchant – even when colluding with the registration center– cannot identify a buyer (anonymity). Furthermore, the merchant is not able to tell whether two purchases were made by the same buyer (unlinkability).

Protection of innocent buyers: No coalition of buyers, the merchant, and the registration center is able to generate a proof $\tilde{p}$ such that $\mathsf{FVer}(ID_B, y_C, \tilde{p}) = 1$, if buyer ID_B was not present in the coalition.

Revocability and collusion resistance: Any collusion of up to c buyers aiming at producing a version $\hat{D} \in \mathcal{D}$ from which none of them can be re-identified will fail: from $\hat{D}$ the merchant will obtain enough information to identify at least one collusion member.

2.3 The protocol

Assume that P^0 has a content D_0 to be multicast and fingerprinted for each receiver. Let $P^1, \cdots, P^N$ be the receivers interested in that content. Then the following protocol can be used to distribute the fingerprinting task.

Protocol 1 (Distributed multicast fingerprinting)

1. *P^0 and P^1 run an anonymous fingerprinting scheme conforming to the model described in Section 2.2.3 and having the following features: i) the underlying watermarking is* blind*; ii)* FRec *is a protocol which needs the help of the registration center (who is assumed to be trusted). After running* FReg *and* FPrint*, P^1 obtains a fingerprinted copy D_{01} of the content and P^0 obtains a transaction record $t_{0,1}$ (partially or totally consisting of information input by P^0 herself like the transaction number).*

2. **For** $i := 1$ **to** $N - 1$*:*

   (a) *P^i and P^{i+1} engage in the same anonymous fingerprinting scheme described above whereby P^{i+1} obtains a fingerprinted version $D_{012\cdots(i+1)}$ and P^i obtains a transaction record $t_{i,i+1}$ (partially or totally consisting of information input by P^i herself like the transaction number);*

   (b) *P^i sends $t_{i,i+1}$ to P^0.*

Some observations on Protocol 1 are in order:

- The need for blind watermarking and for P^i to return the transaction record to P^0 are justified in Section 2.3.1 below. Also, we justify, in that same section, that FRec has to be a protocol needing the collaboration of a trusted registration center, rather than an algorithm run by P^0 alone.

- Each player P^i may engage in anonymous fingerprinting with additional players other than P^{i+1}. However, for the sake of simplicity and without loss of

generality, we ignore such additional transmissions in the above protocol. In the protocol above the multicast tree for the N receivers has depth N: it is actually a line.

- We are implicitly assuming that the underlying watermarking scheme used to embed the fingerprints is such that N or more successive fingerprints can be embedded in D_0 in such a way that:
  1. It still holds that the resulting $D_{01\cdots N}$ is close to D_0, that is $D_{01\cdots N} \in \mathcal{D}$.
  2. Embedding a new fingerprint does not destroy the previously embedded fingerprints. In fact, this results from the aforementioned marking assumption and the previous assumption on the "closeness" of D_0 and $D_{01\cdots N}$.
- We are not assuming any control on the value of N by P^0. In line with the previous remark, we assume that the depth N will not grow to the point of causing $D_{01\cdots N} \notin \mathcal{D}$ This is a self-enforcing policy: no one is interesting is a perceptually bad version of the content. If the number N of players interested in the content turned out to be greater than the number N' of successive embeddable fingerprints, a possible solution is to use a multicast tree for the N receivers whose depth is $N' < N$. This means that some players engage in anonymous fingerprinting with more than one other player.

Note 1 (On content payment) Our protocols do not explicitly consider payment by the content receivers to P^0. Our main focus is on fingerprinted multicast rather than on content sale. However, an easy way to force the receivers to pay for the received content would be to encrypt parts of it: the receivers would then need to pay to P^0 to get the decryption key. Payment by each receiver could be anonymous (*e.g.* [13]) and it could specify a receiver's temporary alias e-mail address to which P^0 should send the decryption key. Payment to P^0 could be sent by each receiver P^i together with a transaction number tn_i provided to P^i by P^{i-1} and obtained by P^{i-1} as a one-way hash function of the transaction record $t_{i-1,i}$. In this way, P^0 would be able to associate each payment with a particular transaction record. Since a key is much shorter than the content, sending a key in unicast to each receiver should not pose bandwidth problems at P^0. Of course, partial content encryption means that fingerprinting during the redistribution chain would have to be limited to the unencrypted parts: ciphertext cannot be fingerprinted, because doing so would render decryption impossible. If a receiver P^i leaked her received key to P^{i+1}, then the latter player and all players P^j with $j \geq i$ would be able to decrypt the content for free. However, the fact that the receivers stay anonymous to each other discourages this colluding behavior: P^i has no particular incentive to leak her key to unknown peers. Furthermore, if P^j uses a leaked key to unlawfully decrypt the content, her penalty in case of redistribution will increase (see Note 2 below); hence, skipping payment at least strengthens redistribution avoidance by peers.

2.3.1 Security and privacy analysis

We define security in Protocol 1 as the ability of P^0 to trace the redistributor in case of detecting unlawful redistribution of the content.

When P^0 detects a redistributed copy $\tilde{D} \in \mathcal{D}$, all P^0 needs to do is to run the following protocol.

Protocol 2 (Redistributor identification)

1. *Let* $T = \{t_{0,1}, t_{1,2}, \cdots, t_{N-1,N}\}$ *be the set of transaction records received by* P^0 *after Protocol 1.*
2. *Set* $i := N - 1$ *and* $recovered :=$ **false**.
3. **While** $recovered =$ **false do**
   (a) *Run, with the registration center, the* FRec *protocol with inputs the redistributed content* $\tilde{D}$ *and* $t_{i,i+1}$.
   (b) **If** *the identity of* P^{i+1} *can be successfully recovered* **then** $recovered :=$ **true else** $i := i - 1$.
4. *Output the identity recovered for* P^{i+1} *as the redistributor's identity.*

Some remarks on the above identity recovery process follow:

- The anonymous fingerprinting scheme used must be based on an underlying blind watermarking method. Indeed, by construction of Protocol 1, unless the redistributor is P^1, P^0 does not know the original unmarked content corresponding to $\tilde{D}$. Imagine that the redistributor is P^{i+1} with $1 \leq i < N$. In that case, $\tilde{D} = D_{01\cdots(i+1)}$ and the original unmarked content is $D_{01\cdots i}$, only known to P^i, not to P^0.
- P^0 wants to obtain the identity of the *last* player who fingerprinted the redistributed content: trying first P^N, then P^{N-1}, and so on, ensures that P^0 will only need to obtain the identity of *one* player, namely the dishonest one. Also, since FRec needs the help of the trusted registration center, the latter can be trusted to help P^0 in recovering only one identity; this preserves the privacy of honest players. Trust here is important, because an untrusted registration center could not just reveal more than one identity, but also frame honest buyers by revealing their identities as if they were redistributors. A way to relax the trust assumption is to use several registration centers rather than one; then the majority answer they give in FRec is probably right: if most registration centers are honest and they coincide in accusing a certain buyer, this buyer is probably the dishonest one.
- Protocol 2 requires to start the search from the most recent transaction record and proceed backwards. So P^0 must have some way to determine the order of the transactions. A first approach could be for P^0 to store the reception time for

each of the transaction records. The use of reception times can lead to errors if the real ordering of the transactions does not match the order of reception of the transaction records. However, if an honest receiver P^j is deemed guilty, she will be able to prove her innocence by showing a transaction record $t_{j,j+1}$ that, together with $\tilde{D}$, allows the registration center to recover the identity of P^{j+1}. Another approach to avoid misidentifications would be to require P^i to append a timestamp to the transaction record $t_{i,i+1}$ so that P^0 could reconstruct the transaction ordering.

In Protocol 2 P^0 has in principle all transaction records, because, if Protocol 1 is correctly followed, after anonymous fingerprinting between P^i and P^{i+1}, P^i sends the resulting transaction record to P^0, for all i. However, Protocol 2 works even if some peers fail to send the transaction record to P^0 in Protocol 1. Assume that P^i and P^{i+1} engage on a transfer of content and that this transfer takes place without any fingerprinting or without P^i sending the resulting transaction record to P^0. Since P^0 does not have the transaction record for the transfer to P^{i+1}, P^0 will not be able to identify P^{i+1} in case P^{i+1} performs unauthorized redistribution. By following the chain of transaction records upwards, at some time P^0 will test if the redistributed copy found can be related to P^i. Hence, P^0, together with the registration center, will be able to obtain P^i's identity and P^i will be found guilty. Note that P^i is indeed guilty for not having correctly followed Protocol 1 and thus can be held liable for the redistribution performed by the (anonymous) P^{i+1}.

Not sending the transaction record to P^0 is therefore risky. A peer P^i who acts this way is implicitly accepting liability for any potential unathorized actions performed by any peer down the chain, that is any peer P^j with $j > i$. If there is another peer $P^{j'}$ with $j' > i$ sending a transaction record to P^0, then P^i will only be held liable for what peers P^j with $i < j < j'$ do.

Note 2 In case of content payment (see Note 1 above), if all receivers are honest, P^0 should receive payment associated to every transaction record. Let us assume that some receiver P^j obtains the decryption key without having paid for it. In this case, P^0 will not have any payment associated to $t_{j-1,j}$. Up to here, no action can be taken by P^0: first, because P^j is anonymous; second, because P^0 cannot prove that P^j actually decrypted the content. However, imagine further that a *decrypted* re-distributed copy is detected by P^0 which leads to identification of P^j using Protocol 2. In this case, P^0 can take action against P^j based on a double offense: illegal redistribution and lack of payment.

2.4 Rational involvement of players: co-utility

In this section, we show how to motivate the players in Protocol 1 to rationally play their corresponding roles as specified in the protocol. Showing that players have no interest in deviating is especially necessary in a peer-to-peer (P2P) protocol whose correct operation depends on the commitment of peers $P^1, \cdots, P^N$.

P^0 has an obvious interest in following Protocol 1. If she deviates from the protocol by not correctly participating in the anonymous fingerpriting at Step 1, the entire distributed multicast fingerprinting does not even start. Let s^0 the strategy whereby P^0 follows the protocol.

Each peer $P^i \in \{P^1, \cdots, P^N\}$ is assumed to be interested in getting the content; therefore, she will not deviate from correct anonymous fingerprinting with P^{i-1}. However, $P^i \in \{P_1, \cdots, P^{N-1}\}$ has at least four possible strategies with respect to P^{i+1}:

s_0^i: Correctly follow Protocol 1 by engaging in anonymous fingerprinting with P^{i+1} and returning transaction record $t_{i,i+1}$ to P^0;

s_1^i: Deviate from Protocol 1 by engaging in anonymous fingerprinting with P^{i+1} but *not* returning $t_{i,i+1}$ to P^0;

s_2^i: Deviate from Protocol 1 by not engaging in anonymous fingerprinting with P^{i+1} but returning a fake transaction record $t_{i,i+1}$ to P^0.

s_3^i: Deviate from Protocol 1 by not engaging in anonymous fingerprinting with P^{i+1} and not sending any transaction record.

We next go through an exercise of mechanism design (see Chap. 23 of [30]), to find how Protocol 1 needs to be modified to ensure that, for any player P^i, her rational choice is strategy s_0^i.

2.4.1 Utility without reward or punishment

Consider the following payoffs:

d_i: Payoff that P^i derives from obtaining $D_{0\cdots i}$ *without losing her anonymity*. That is, d_i combines the functionality payoff of P^i obtaining the content and the privacy payoff of P^i preserving her anonymity thanks to anonymous fingerprinting with P^{i-1}. If P^i pays a fee or reward for obtaining the content, this fee or reward must be subtracted from the previously defined payoff to get the remaining d_i.

$-v_i$: Negative payoff (that is, cost) that P^i incurs from engaging in anonymous fingerprinting with P^{i+1}; this cost may be quantified in terms of computation and communication.

$-w_i$: Negative payoff that P^i incurs from returning the transaction record $t_{i,i+1}$ to P^0; this cost would correspond to the communication cost of sending the transaction record.

If there are no other payoffs (like reward earned for following the protocol or punishment incurred for not following it), the general utility functions of the above strategies are the following:

$$\mathbf{u}_i(s_0^i) = d_i - v_i - w_i$$

$$\mathbf{u}_i(s_1^i) = d_i - v_i$$

$$\mathbf{u}_i(s_2^i) = d_i - w_i$$
$$\mathbf{u}_i(s_3^i) = d_i$$

Clearly, strategy s_3^i has the maximum utility. In these conditions, the dominant strategy solution of the game is

$$(s^0, s_3^1, -, \cdots, -)$$

In plain words, the rational equilibrium is for P^0 to start the distributed fingerprinting and for P^1 to acquire an anonymously fingerprinted D_{01} and exit Protocol 1. Strategies by P^2 to P^N are irrelevant, because their participation in the protocol is prevented by P^1's choice of strategy s_3^1. Clearly, this dominant solution means that players are not rationally interested in correctly following the protocol.

2.4.2 Utility with reward and no punishment

In an attempt to induce rational players to correctly follow Protocol 1, we can think of introducing a reward for a player who forwards the content to other players. There are two ways to reward player P^i:

Centralized reward: After engaging in anonymous fingerprinting with P^{i+1}, P^i returns the transaction record $t_{i,i+1}$ to P^0 and gets a reward $r_{i,i+1}$ from P^0.

Distributed reward: After engaging in anonymous fingerprinting with P^{i+1}, P^i gets a reward $r_{i,i+1}$ from P^{i+1}, who discounts $r_{i,i+1}$ from her payoff d_{i+1}.

It is not difficult to see that the centralized reward has at least two serious problems:

- P^0 bears all the costs of the rewards. Therefore, if P^0 is selling the content to make a profit, P^0 needs to charge a substantial fee to P^1, her only direct buyer.
- Under the centralized reward, there is incentive for P^i to cheat by playing strategy s_2^i: return a fake transcript to P^0 without actually engaging in anonymous fingerprinting with P^{i+1}.

Hence, the distributed reward seems clearly preferable. In this case, the utility functions of the four strategies of P^i are:

$$\mathbf{u}_i(s_0^i) = d_i + r_{i,i+1} - v_i - w_i$$
$$\mathbf{u}_i(s_1^i) = d_i + r_{i,i+1} - v_i$$
$$\mathbf{u}_i(s_2^i) = d_i - w_i$$
$$\mathbf{u}_i(s_3^i) = d_i$$

If the reward is sufficient to cover the costs of P^i engaging in anonymous fingerprinting with P^{i+1}, that is, if $r_{i,i+1} \geq v_i$, then s_1^i has the maximum utility. In these conditions, the dominant strategy solution of the game is

$$(s^0, s_1^1, \cdots, s_1^{N-1}, s_3^N)$$

In plain words, the rational equilibrium is for P^0 to start the distributed fingerprinting and for P^i ($i = 1, \cdots, N-1$) to acquire and forward a fingerprint to P^{i+1}. The strategy of P^N can only be s_3^N, because P^N is not supposed to forward the content any further.

We have achieved some improvement: players are rationally interested in multicast fingerprinting, but they do not report the transaction records to P^0, which hampers redistributor identification by P^0.

Note 3 (Implementing reward payment) A technical issue is how to implement the payment of distributed rewards. Rewards must be paid between players who are anonymous to each other. This precludes the use of P2P payment techniques requiring peer identification, like the cascaded payments proposed in [6]. Workable alternatives are anonymous micropayments between players. The anonymous version of the PayWord scheme described in [41] can be used, for example. The payer sends an initial coupon T_0 of a hash chain (the payword) to the payee, where T_0 has been blindly signed by the payer's bank; then the payer reveals a certain number of successive coupons of the payword (where T_i is a hash pre-image of T_{i-1}) in order to adjust to the amount of the reward to be paid. Double-spending detection mechanisms can be added to the signature on T_0 that cause the payer's anonymity to be lost if he uses the same payword twice (*e.g.* see [13]). If a payer must reward several times the same payee (such a situation can be detected even if players are anonymous to each other, *e.g.* using a cookie mechanism), the payer can keep sending to the payee successive coupons of a payword whose T_0 was exchanged and verified by the payee in a previous transaction. Doing so has the advantage of amortizing over several transactions the computation associated to producing and verifying the signature on T_0.

2.4.3 Utility with reward and punishment

A punishment mechanism can be added as an incentive for players P^1 through P^{N-1} to return transaction records to P^0.

Let $-p_i$ be the expected negative payoff (punishment) that P^i incurs when accused of redistribution as a result of not having returned a valid transaction record $t_{i,i+1}$. This is actually an *expected* negative payoff, computed as the probability of being accused times the cost of being accused. This negative payoff includes the loss of anonymity (as a result of the redistributor identification algorithm) and may include fines or other penalties (which are easy to apply after anonymity loss).

Under strategy s_1^i, P^i fingerprints the content but she does not return the transaction record. Under s_2^i, P^i forwards the content without fingerprinting and returns a fake transaction record. Under s_3^i, P^i forwards the content without fingerprinting or returning any transaction record. Therefore, in none of those three strategies is a valid transaction record returned, hence all of them incur the punishment. Note that, when running the redistributor identification protocol (Protocol 2), a fake transaction record is treated like a non-existing transaction record: if $t_{i-1,i}$ is the last authentic transaction record received by P^0, then no matter whether P^i sent a fake $t_{i,i+1}$ or no record at all, P^i will be accused of redistribution and hence punished.

We can now recompute the utilities of the four strategies available to P^i:

$$\begin{aligned} \mathbf{u}_i(s_0^i) &= d_i + r_{i,i+1} - v_i - w_i \\ \mathbf{u}_i(s_1^i) &= d_i + r_{i,i+1} - v_i - p_i \\ \mathbf{u}_i(s_2^i) &= d_i - w_i - p_i \\ \mathbf{u}_i(s_3^i) &= d_i - p_i \end{aligned} \tag{2.2}$$

Assume like above that $r_{i,i+1} \geq v_i$. Also, assume that $-p_i \leq -w_i$, that is, that not returning the transaction record is worse than returning it. With those assumptions,

$$\mathbf{u}_i(s_2^i) \leq \mathbf{u}_i(s_3^i) \leq \mathbf{u}_i(s_1^i) \leq \mathbf{u}_i(s_0^i)$$

so that s_0^i is the strategy with maximum utility. Hence, the dominant strategy solution of the game is

$$(s^0, s_0^1, \cdots, s_0^{N-1}, s_3^N)$$

In plain words, the rational equilibrium is for P^0 to start the distributed anonymous fingerprinting and for P^i ($i = 1, \cdots, N-1$) to correctly follow Protocol 1. The strategy of P^N can only be s_3^N, because P^N is not supposed to forward the content any further.

With the proposed modifications, we have succeeded in inducing a rational behavior in the players that causes them to correctly following the intended multicast fingerprinting protocol.

Lemma 1 *With the utility functions defined in Equations (2.2), there is co-utility between P^i and P^{i+1} for $i \in \{0, \cdots N-1\}$.*

Proof *With the utilities in this section, the dominant strategy solution has been shown to be the one in which every player P^i plays s_0^i. Note that s_0^i is precisely the strategy which yields the best possible payoff d_{i+1} for P^{i+1}: indeed, P^{i+1} obtains the content while preserving her anonymity (thanks to anonymous fingerprinting). Now, whatever the strategy chosen by P^{i+1}, the utility function $\mathbf{u}_{i+1}$ monotonically increases with d_{i+1}.*

Hence, the best strategy for P^i results in enhanced utility for P^{i+1}, whatever P^{i+1}'s strategy. The lemma follows. □

2.5 Proof of concept

The objective of this section is to provide a proof of concept to show that the proposed protocol can be put into practice with existing watermarking technologies. This implies that the results of this chapter are not merely theoretical: a practical application of the discussed protocol is implementable.

The protocol described in Section 2.3 has been realized using the audio watermarking scheme described in [33]. This watermarking scheme can be used as a building block for anonymous fingerprinting and it satisfies the requirements listed in Section 2.3. The scheme is blind, so that it is possible to extract the embedded mark from

a marked audio object without knowing the original unmarked audio object. Also, the scheme tolerates embedding several successive fingerprints without significant damage to the content utility or the previous fingerprints.

The scheme uses a double embedding strategy:

- A time-domain synchronization watermark ("SYN") is embedded for fast search of the information watermark position;
- A frequency-domain information watermark is embedded next to the SYN marks.

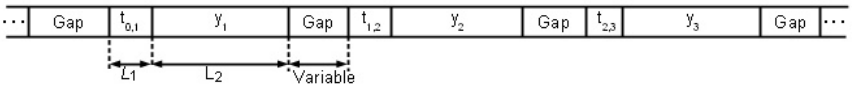


Figure 2.1: Embedding strategy

This double embedding strategy makes it possible to embed the transaction records $t_{i,i+1}$ and the receiver related information y_B in different domains, as depicted in Figure 2.1. The transaction records can be embedded as synchronization marks in the time domain with different bit strings, and the related information y_B can be embedded more robustly in the frequency domain. This scheme has the additional advantage of a very fast search of transaction records; extracting an embedded transaction record from a portion of audio takes less time than playing that portion.

In order to preserve anonymity and make the registration center necessary for redistributor identification (as mandated by Protocol 2), we let y_B be the receiver identity encrypted under the registration center's public key. To obtain unlinkability, a random nonce is appended to the receiver's identity before encrypting it under the registration center's public key. Embedding, next to y_B, a hash of x_B (or x_B encrypted with the public key of the receiver) has the additional advantage of thwarting a collusion of the sender P^0 and the registration center, who would not be able to produce a correctly fingerprinted copy of the content corresponding to any receiver.

If the sender P^0 finds a version of the audio file illegally redistributed on the Internet, she can search for the transaction records in the time domain (fast search) and then extract the information y_B related to the malicious receiver. This information (y_B) will then be sent to the registration authority in order to identify the illegal redistributor.

Within this framework, two different experiments have been performed for a set of six players: P^0 (sender) and P^1, P^2, P^3, P^4, P^5 (receivers). Two approaches have been compared:

Centralized unicast: The sender and each receiver separately engage in an anonymous fingerprinting protocol to generate different fingerprinted copies D_{01}, D_{02}, D_{03}, D_{04} and D_{05} from the original content D_0.

Distributed multicast: The sender P^0 and the receiver P^1 engage in an anonymous fingerprinting protocol to generate D_{01} from the original content. P^0 generates

the transaction record $t_{0,1}$ to be embedded in the time domain. Subsequently, P^i and P^{i+1}, for $i = 1, 2, 3, 4$ engage in the same fingerprinting protocol to generate D_{012}, D_{0123}, D_{01234} and D_{012345}; the corresponding transaction records $t_{i,i+1}$, for $i = 1, 2, 3, 4$ are returned to the sender by P^i (who plays the merchant role in the transaction between P^i and P^{i+1}). P^0 keeps a sorted list of the transaction records. Each copy of the digital content carries the fingerprints corresponding to all previous receivers.

In order to preserve the privacy of the input and output information ($t_{i,i+1}$, $D_{01\ldots i}$, y_{i+1}, $D_{01\ldots i+1}$) in each execution of the fingerprinting scheme, a secure two-party computation protocol, as those presented in [14, 8], is required as a building block of the anonymous fingerprinting protocol. In the distributed multicast protocol, only P^0 has access to the the original content D_0, whereas only P^1 knows the information y_1 which is embedded in D_{01}. The same applies for the subsequent executions of the fingerprinting protocol. This secure multiparty computation approach introduces an additional overhead which can be shown to be poly-logarithmic in the number of receivers and the size of the circuit (or algorithm) needed to implement the scheme [16].

The original content (D_0) used in the experiments is the 30-second violoncello file ("vioo10_2.wav") available from the Sound Quality Assessment Material corpus [5]. In the centralized unicast protocol, the file is divided into 10-second segments and an instance of the fingerprint is embedded into each segment. The fingerprint consists of an information watermark (y_B) embedded in the FFT domain preceded by a transaction record embedded in the time domain (see Figure 2.1).

It must be pointed out that the watermarking scheme [33] allows embedding a long bit string. In addition, to enable even longer embedding capacity, the information could be encrypted, sent to the merchant, and the key could be embedded instead of y_B. Using the parameters specified in the experimental section of [33], each instance of the watermark requires 1.30 seconds of audio. Hence, each 10-second segment allows the inclusion of up to 7 different watermarks, more than enough for the 5 levels of embedding considered in the experiments reported here.

With a longer audio content, the segments could be chosen long enough to allow for, say, 10 different watermarks. In that case, if every receiver could engage in the fingerprinting protocol with up to f other receivers, the number of potential receivers could increase in powers of f at each step; in the 10-th step, up to f^{10} receivers would be reached. It is easy to see that a value $f = 9$ would be enough to cover a number of receivers equal to half of the earth's population.

The centralized unicast and distributed multicast protocols above have been compared in terms of: i) CPU time and bandwidth required from the sender P^0; and ii) the transparency of the resulting fingerprinted content.

In what regards CPU time, the fingerprinting scheme has been tested in a Matlab (interpreted) implementation on a 3.0 GHz (single core) Pentium IV processor with 1 GB of RAM. The overhead of the two-party secure computation scheme has not been taken into account in this simulation, but it can be reckoned to multiply by a constant greater than 1 the CPU time needed for two parties to complete an anonymous fingerprinting; hence this overhead does not influence the following comparison between the centralized unicast and the distributed multicast. In the centralized unicast approach,

Table 2.1: Transparency results

Protocol	Content	# fingerprints	ODG
Centralized	D_{01}	1	0.000
	D_{02}	1	0.000
	D_{03}	1	0.000
	D_{04}	1	0.000
	D_{05}	1	0.000
Peer-to-peer	D_{01}	1	0.000
	D_{012}	2	-0.004
	D_{0123}	3	-0.034
	D_{01234}	4	-0.115
	D_{012345}	5	-0.193

the sender P^0 uses 9.81 seconds of CPU time to produce the 5 marked copies of the content ($D_{01}, \ldots, D_{05}$) and transmits 26,519,020 bytes of information (5 different versions of the uncompressed audio file). In the distributed multicast, the sender needs to run the fingerprinting scheme just once (with the receiver P^1) taking 1.97 seconds of CPU time and sending 5,303,804 bytes. Hence, from the sender's point of view, distributed multicast consumes just a 20% of CPU time and bandwidth compared to centralized unicast. Since the sender is the bottleneck in centralized unicast, the saving allowed by distributed multicast is relevant.

Regarding transparency, the Objective Difference Grade (ODG) based on the ITU-R Recommendation standard BS 1387 [25, 43] has been used. This standard makes it possible to evaluate the transparency of the fingerprinting scheme by comparing the perceptual quality of the marked files with respect to the original content D_0. The ODG values are in the range $[-4, 0]$, where 0 means imperceptible, -1 means perceptible but not annoying, -2 means slightly annoying, -3 means annoying and -4 means very annoying. In order to evaluate the ODG, we have used the Opera software by Opticom [3].

The imperceptibility results are shown in Table 2.1 for all the files obtained with both the centralized unicast and the distributed multicast. As it can be noticed, the transparency of the five files resulting from the centralized protocol is perfect (ODG $=$ 0), whereas it slowly decreases for each successive receiver in the distributed multicast protocol. However, even with 5 embedded fingerprints, the ODG result is much closer to 0 (imperceptible) than -1 (perceptible but not annoying); hence, even in this worst case, the perceptual quality achieved by the distributed multicast protocol can be regarded as very satisfactory.

Now let us imagine that the receiver P^3 decides not to return the transaction record $t_{3,4}$ to the sender P^0. In this case, if P^0 finds an illegal redistribution of file $D_{012\ldots m}$, P^0 will send the redistributed file and $t_{2,3}$ to the registration center to track the liable receiver, because $t_{2,3}$ is the last transaction record available to P^0. Hence, P^3 will be held guilty of illegal redistribution due to her decision of not returning $t_{3,4}$ to the

sender.

2.6 Conclusions and future research

We have described a peer-to-peer protocol for distributed multicast of fingerprinted content which has the interesting properties that:

- Each receiver obtains a different fingerprinted copy of the content which allows the sender to trace redistributors;
- The sender does not need to prepare and send a separate fingerprinted copy to each receiver, so that its computational and bandwidth burden is equivalent to the case of there being a single receiver;
- Receivers rationally co-operate in a peer-to-peer fashion thanks to a system of rewards and punishments which ensures that each receiver's best strategy is to loyally follow the prescribed peer-to-peer multicast protocol; in this respect the protocol is said to be co-utile.

Future research will investigate applications of the proposed peer-to-peer multicast protocol to scenarios other than redistribution control, such as enforcing expiration dates on data items in view of digital forgetting.

Chapter 3

Privacy-Aware Peer-to-Peer Content Distribution Using Automatically Recombined Fingerprints

3.1 Contribution of this chapter

We propose a P2P distribution scheme [32] of fingerprinted content whereby the merchant originates only a set of M seed copies of the content and sends them to M seed buyers. All subsequent copies are generated from the seed copies. Each non-seed buyer obtains her copy of the content by running a P2P purchase software tool. The copy obtained by each buyer is a combination of the copies provided by her sources (parents). The fingerprint of each buyer is thus a binary sequence formed as the combination of the sequences of her sources. This peer-to-peer distribution scheme makes it possible for the merchant to save bandwidth and CPU time, while still being able to trace unlawfully redistributed content.

In the proposed approach, the fingerprints of the buyers do not need to be registered in any way and, thus, all buyers can preserve their privacy as long as no illegal content redistribution occurs. However, when an illegally redistributed file is found, it must be possible to link its binary sequence to a particular individual (buyer).

To satisfy the above conditional privacy, a P2P proxy (or set of proxies) is used to create anonymous connections between buyers such that each source and destination buyers do not lose their anonymity. The P2P proxy also sends a transaction record to a transaction monitor whenever a buyer obtains fragments of the contents from another buyer. The contents of this transaction record are:

- The usernames (or pseudonyms) of the two buyers participating in the transaction.

- An encrypted hash of the whole fingerprint of each buyer.

The purpose of storing the above transaction records at the transaction monitor is to enable tracing of illegal redistributors. Note that buyers stay anonymous to each other, but only pseudonymous versus the transaction monitor; however, the transaction record does not specify which fragments come from which buyer, so that the privacy of the buyers' fingerprints is preserved. The encrypted hash is used by the authority in case a buyer intends to cheat the tracing system by showing a different (modified or borrowed) copy of the content. Since the transaction monitor only records a hash of the true fingerprint and buyer pseudonyms that are not linked to specific fragments of the content, no coalition of the transaction monitor, the seller or other buyers can be used to frame an innocent buyer (by unjustly accusing her).

In order to carry out an *a posteriori* identification of redistributors, a correlation test is run taking the fingerprint of the illegally redistributed content and the fingerprints of the M seed buyers as inputs; among the seed buyers, the test attempts to determine the maximum-likelihood ancestor of the content.

The fingerprints of the selected ancestor's children are retrieved by the tracing authority (with the collaboration of the buyers) and the maximum-likelihood test is run again with these fingerprints and the traced fingerprint as input. When a match is found between both fingerprints (maximum correlation between fingerprints) the redistributor is identified. If a buyer refuses to take the corrlation test, the hash of the fingerprint can be used as evidence against her. If the hash of a buyer's fingerprint exactly matches the hash of the redistributed content's fingerprint, then the buyer is charged with unlawful redistribution. Otherwise, if the hashes differ, the refusing buyer will be charged with contract breach and the test is performed using the hashes of the fingerprints as a replacement of the entire fingerprints. In addition, the registered hashes of the fingerprints are enough to discourage buyers from cheating the tracing system by using borrowed or altered copies of the contents.

If the correlation test is carried out using a secure multiparty computation approach [14, 16], the exact fingerprint of honest buyers will not have to be revealed (although computing some correlation with it and obtaining the complete hash will be required), but their privacy (the fact that they have purchased the contents) will not be preserved versus the tracing authority. However, buyer privacy with respect to the authority will only be broken for those few users affected by correlation tests and their identity will be revealed only to the identification agent. On the other hand, the privacy of the majority of users is preserved and their fingerprints remain private.

In addition to attractive privacy properties, it will be shown that, in practice, the proposed scheme offers good security properties, namely collusion resistance vs dishonest colluding buyers (if a particular anti-collusion strategy is used) and buyer frameproofness vs a malicious merchant.

The rest of this chapter is organized as follows. Section 3.2 describes the basic principles used in the paper for peer-to-peer distribution of fingerprinted contents. Section 3.3 presents the P2P distribution protocol and how transfers between peer buyers are anonymized. Section 3.4 presents a protocol for tracing unlawful redistributors, together with some examples; a modification of the method is presented to make tracing resistant against buyer collusions. Section 3.5 discusses security assumptions, as

well as the buyer privacy and frameproofness offered by our fingerprinting proposal. Section 3.6 contains simulation results. Finally, Section 3.7 summarizes conclusions and future research issues.

3.2 P2P distribution of recombined fingerprinted contents

The basis of P2P content distribution is that the shared contents are distributed by some users to others. As soon as some fragments of the content are received, destination users become sources for others. A file is thus obtained by joining the fragments of several sources together. Typically, a hash value of the shared content is used by P2P clients to identify files. Two files having the same hash value are considered equal. The upload/download process of a file from different sources is depicted in Figure 3.1. In this figure, the destination obtains fragments from three different sources that are joined together to form the content.

3.2.1 Mating approach for fingerprinting

In this section, we introduce a novel concept of automatic binary fingerprints partly inspired on biological mating and inheritance. The relationship between biology and the scheme in this paper is rather weak and, thus, we refrain from calling our scheme "genetic". However, some biological analogies are highlighted in this section to introduce the basis of the suggested scheme.

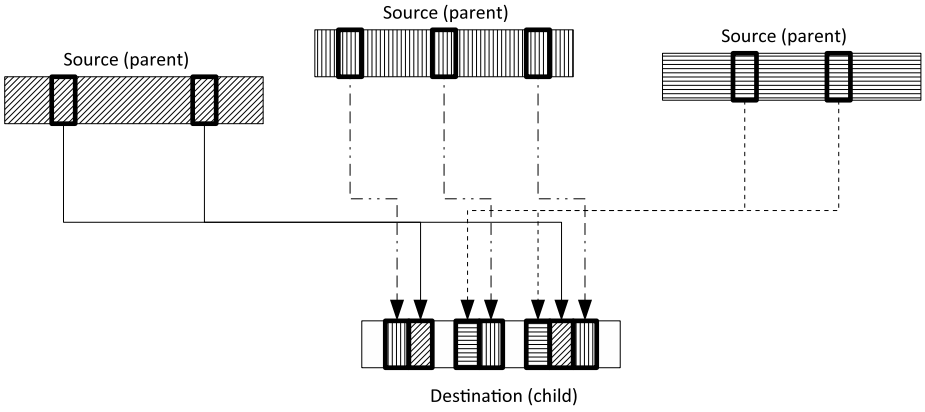


Figure 3.1: Upload/download of the content (mating process)

In this paper, fingerprints are constructed as binary sequences and each bit might be considered as the counterpart of the nucleotides of a DNA sequence. This is similar to the approach taken in Genetic Algorithms [24] for solving optimization problems. Just like DNA sequences are formed by different genes which encode a give protein, the binary fingerprints used in this paper are formed by (fixed-size) segments that may

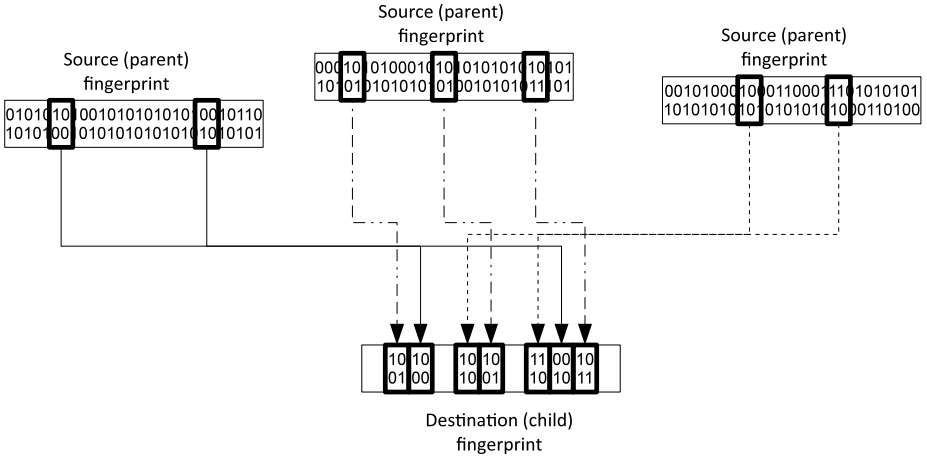


Figure 3.2: Automatic recombined fingerprint construction

be considered as analogs of genes. When a buyer obtains a copy of a P2P-distributed content using some specific software, the binary fingerprint of her copy is a combination of the segments of the sources of the content (referred to as "parents" from the biological analogy). In this case, the number of parents of some content does not have to be exactly two as in the natural world. Hence, the mating process in the suggested fingerprinting scenario must be understood in a generalized sense, not limited to two parents. Fingerprints can be considered as being "automatically generated" from the fingerprints of the parents. Despite this "automatic generation" of fingerprints, the constructed sequences are still valid for identification purposes.

In order to identify the culprit of an illegal redistribution, a search process must be carried out in the P2P distribution graph. This search is performed with the help of a correlation function which tries to minimize the number of explored nodes. The idea is to search for a given fingerprint from "ancestors" to "descendants" in the graph. A simple correlation function between two binary strings can be used to determine the likelihood that a given buyer is an "ancestor" of another one. A path from "ancestors" of a given buyer can thus be formed using this correlation function to identify the source of an illegally redistributed copy of the content.

3.2.2 Requirements on fingerprint embedding

If the binary fingerprints described in Section 3.2.1 are to be found in a P2P-distributed content, an embedding method is to be used for the M seed buyers. It is enough to embed randomly generated fingerprints for the seed buyers such that their pairwise correlation is low. This embedding scheme must fulfill the following conditions:

1. The embedded fingerprint must be a binary sequence spread along the whole content (file). Furthermore, the fingerprint must be separated into pieces which

are embedded into different blocks (or fragments) distributed by the P2P software. These (fixed-sized) pieces of the content contain a full segment of the fingerprint. For example, if the P2P software uses 32-KB (kilobyte) fragments, each segment of the fingerprint should be embedded into one of these fragments and the fingerprint extraction method must be robust against fragmentation in 32-KB units, as long as the beginning and the end of the fragments are respected. This process is illustrated in Figure 3.2.

Note that this is not always possible with non-block-based embedding schemes. An example of block-based audio watermarking system which may be used for fingerprinting in this scenario is presented in [33].

2. Even if the versions of the content obtained by different buyers will not be bit-wise identical (the fingerprints embedded into the buyers' copies will differ as a consequence of the P2P distributed download), these versions should be "perceptually" identical, because the distributed content must have the same high quality for all buyers. This means that a standard hash function which produces different hash values even after a single bit change would not be useful in this application. A perceptual hash function [15] for which the same hash value is obtained for different (perceptually identical) versions of the same content would be required if hash values are used for indexing in the P2P distribution software.

If the previous two conditions hold, fingerprinting occurs in an automatic way as contents are obtained by buyers from different sources. No additional overhead for embedding is required. Note that the above automatic fingerprinting requires more than one content source for each buyer to exist: in case of a single source, the fingerprint would be identical for both the source and the buyer. Although some segments of the fingerprint could be modified by running the embedding method in these buyer-to-buyer transfers, this would reduce the simplicity of our proposal. The simplest solution is to enforce at least two parents for each buyer, and this is the choice made in this paper.

3.2.3 Coprivacy in parent-child relationship

If it can be enforced that there be at least two parents for each buyer, this is the simplest and most effective solution, because, as said above, fingerprinting is automatic in this case. Fortunately, it turns out that it is in the selfish interest of a child buyer to obtain her content from more than one parent, and it is in the selfish interest of a parent buyer to split her content into more than one child buyer:

- If a child obtains her entire content from a single parent then, her fingerprint will be the same as her parent's fingerprint. Then, if the parent happens to illegally redistribute the content, the child risks being unjustly accused of redistribution (see Section 3.4 below). Obtaining the content from several parents is a simple and automatic way to avert that risk.

- If a parent sends her entire content to a single child, her child will inherit the parent's fingerprint. Hence, if the child happens to illegally redistribute the content,

the parent risks being unjustly accused of redistribution. Splitting the content among several children is the best option for the parent.

This situation in which the best strategy to preserve one's own privacy is to act in such a way that someone else's privacy is protected is known as coprivacy [19, 20]. In game-theoretic terms, the vector of strategies $(\text{multiple children}, \text{multiple parents})$ is a Nash equilibrium between parent and child.

The coprivacy property ensures that, whenever a child buyer can obtain her content from more than one parent, she will do so; it also ensures that parents will be interested in not passing their entire content to a single child buyer. The latter condition can be easily enforced by the P2P distribution software. For example, when a parent is sending the content through a proxy, it can be enforced that the connection be closed as soon as a given threshold fraction (*e.g.* 50 or 60%) of the content has been sent. The software can also block any further attempt by the proxy to establish a connection with the same parent for the same content (for some given time window). Each proxy should be forced to choose at least two different parents.

3.2.4 Building blocks and notation

In order to design protocols for the different steps of the distribution system, the following building blocks are required:

- Public-key cryptography is required in different steps below. Let $E(\cdot, K)$ be the encryption function using the public key K and $D(\cdot, K^s)$ be the decryption function using the private key K^s, required to decrypt a content encrypted using E and K, *i.e.* $D(E(x, K), K^s) = x$.
- In particular, the transaction monitor uses the following pair of public and private keys: (K_c, K_c^s). Also, each peer node in the network is supposed to have a public key and a private key.
- For each segment of the fingerprint g_i, a hashing function h produces a 1-bit hash $h(g_i)$. Let h_f be the (ordered) concatenation of the hashes of all segments, called "fingerprint's hash" hereafter. Hence, h_f is constructed as

  $$h_f = h(g_1)|h(g_2)|\ldots|h(g_l),$$

  where l is the number of segments of the fingerprint and "|" stands for the concatenation operator.
- An extraction function exists to obtain the fingerprint from a content. This function receives, as input parameters, the fingerprinted content and a secret extraction key K^e only known by the merchant. This key will be required by the authority to trace an unlawful distribution.

3.3 The P2P distribution protocol

To bootstrap the system, a few seeds of the fingerprinted content must be produced. The proposed approach is for the merchant to produce a small number M of instances

of the content with different pseudo-random binary fingerprints, using some scheme satisfying the conditions described above. These M seeds could be the first buyers of the content who will be the ones contacted by second-generation buyers to obtain further copies of the content. Either the merchant or some trusted authority will keep the association of the first M fingerprints with the identities (or maybe some pseudonym) of the first M buyers. After the system is bootstrapped in this way, all future transactions occur without any further execution of the embedding scheme. Furthermore, all fingerprints from buyer $M + 1$ to the final one are completely anonymous (accessible only if the buyer provides her copy of the content for fingerprint extraction) and do not relate to the buyers' identities. Note that this way of achieving anonymous fingerprinting is much simpler than the anonymous fingerprinting proposals in the literature [37, 10, 8, 18], predicated on some sort of complex cryptographic protocol for *every* transaction. Only the transaction monitor keeps a record of the engaged transactions in case they need to be used in future correlation tests.

We can summarize the P2P distribution protocol as follows.

Protocol 3 (P2P distribution)

1. *For $i := 1$ to M, the merchant generates the i-th seed copy with a random fingerprint embedded in it (the fingerprints of the M copies should have low pairwise correlations).*

2. *For $i := 1$ to M, the merchant forwards the i-th seed copy to the i-th seed buyer. If the seed buyers are genuine rather than dummy buyers, this step can be anonymized as explained below.*

3. *For $i := M+1$ to N, the i-th buyer obtains her copy of the content by composing fragments obtained from a set S_i of parent nodes such that*

$$S_i \subseteq \{B_1, \cdots, B_{i-1}\}$$

and $|S_i| > 1$, where $|\cdot|$ is the cardinality operator and B_j refers to the j-th buyer. This transaction is performed via a proxy (or a set of proxies) and with an anonymous protocol (see below). The proxy registers each transaction at the transaction monitor. Since the same parent may be chosen by different proxies, a transaction record for the same parent, child and content may already exist. In that case no new record would be created. When all the fragments have been transferred for a buyer, the whole fingerprint's hash is also stored in the transaction monitor.

The transaction record stored at the transaction monitor is formed by the following information:

- Username (pseudonym) of the parent (source) buyer.
- Username (pseudonym) of the child (destination) buyer.
- Content hash (used for indexing in the content database).
- Encrypted hash of the child buyer's fingerprint.

- Transaction date and time (for billing purposes).

Note that the transaction monitor does not store the true identities of the buyers, only pseudonyms. Only the merchant has access to the buyers' database, which relates a given pseudonym to real identity data.

The hash of the fingerprint is not stored as cleartext in the transaction monitor, but encrypted under the public keys of the parents and the transaction monitor; the proxy records one encrypted version under each parent's public key. In this way, in case of an investigation, the transaction monitor will need the cooperation of one parent to decrypt the hash. This provides additional anonymity and protection to buyers.

In order to protect the buyers' anonymity, the transfer between buyers must remain anonymous. Otherwise, some buyers (the parents of a child) may collude to generate a replica of the content of another buyer and redistribute it illegally. For a set of fragments, the P2P software runs the following protocol:

Protocol 4 (Anonymous content transfer between buyers)

1. *The child buyer's P2P client software contacts a proxy and requests a group of fragments.*

2. *The proxy selects at least two other buyers (parents) as the sources of the content fragments. This guarantees that each buyer will have at least two parents. The proxy uses an onion routing-like solution (based on Chaum's mix networks [12]) such that the fragments are transferred anonymously from parent to child. Note that the content does not need to be encrypted using public-key cryptography. A one-time symmetric session key can be chosen by the child buyer and be transmitted to the proxy. This session key is used to encrypt the actual fragments, so that the routers cannot see them in cleartext.*

3. *The proxy informs the transaction monitor about the transaction when all fragments have been transferred from parent to child. A transaction record is then stored in the transaction monitor for this parent-child-content association. The proxy also informs the transaction monitor about the number of fragments transferred to the child.*

4. *For each fragment, the proxy also receives the hash of the corresponding segment (the part of the fingerprint embedded in the fragment). Note that a parent does not have any motivation to cheat about the hash bit, since: i) doing so would only favor an unknown child; ii) if she cheats, she may be discovered in future investigations and be accused of contract breach. Additional security based on signatures for the hashes can be easily introduced.*

5. *When the child has received all the fragments of the complete content, the transaction monitor can contact all proxies involved in the transfer and construct the hash h_f of the fingerprint by joining all the segment hashes together, following the steps detailed below. If a buyer chose p proxies, then:*

    - *Each proxy obtains a fragment h_{f_i} of the fingerprint's hash h_f for $i = 1, \ldots, p$ (containing several bits corresponding to the hashes of various*

segments). For simplicity of notation and without loss of generality, it is assumed that the fragments of the fingerprint's hash are consecutive and ordered with respect to the index i: $h_f = h_{f_1}|h_{f_2}|\ldots|h_{f_p}$. Note that a simple permutation of the different hash fragments can be used to make the previous assumption hold.

- *All proxies exchange their fragments of the fingerprint's hash encrypted with the public key of the transaction monitor (K_c). Hence, all proxies have*
$$E_h = E(h_{f_1}, K_c)|E(h_{f_2}, K_c)|\ldots|E(h_{f_p}, K_c).$$
This also means that no single proxy has access to the complete cleartext of the fingerprint's hash.
- *Let $P_{i,j}$ be the j-th parent chosen by the i-th proxy, and $K_{i,j}$ her corresponding public key. For every parent j chosen by the i-th proxy, the proxy sends $E(E_h, K_{i,j})$ to the transaction monitor.*

In this way, only a collusion formed by all the proxies of a child buyer (and possibly the transaction monitor) can replicate the entire fingerprint of the child. If every buyer chooses enough proxies for each content, such a collusion is so unlikely that it can be neglected.

The hash is encrypted under the public key of each parent and registered once per parent. In this way, the cooperation of only one parent is enough to obtain the decrypted fingerprint's hash. The transaction monitor needs one of the parents to use her private key $K^s_{i,j}$ to obtain:
$$D(E(E_h, K_{i,j}), K^s_{i,j}) = E_h.$$
After that, the transaction monitor can use its own private key K^s_c to decrypt the fingerprint's hash:
$$\begin{aligned} &D(E(h_{f_1}, K_c), K^s_c)|D(E(h_{f_2}, K_c), K^s_c)|\ldots|D(E(h_{f_p}, K_c), K^s_c) \\ &= h_{f_1}|h_{f_2}|\ldots|h_{f_p} = h_f. \end{aligned}$$

Regarding the choice of proxies, possibly the simplest and "most distributed" solution would be that all P2P clients (buyers) can be chosen as proxies by the P2P distribution software. Note that proxies do not have to be buyers of the same content and, thus, this would not break the privacy of buyers. In case that malicious proxies are considered, additional security measures shall be introduced, but this issue is left for the future research.

Note 4 (On payment of content) *Our protocol does not explicitly consider payment by the buyers to the merchant. Our main focus is on fingerprinted multicast rather than on content sale. In any case, since the transactions are stored in the transaction monitor, a periodic invoice can be issued by the transaction monitor to the merchant such that the merchant can charge the buyers' accounts with the corresponding amounts. Note that such invoice does not need to specify particular contents, since only the total amounts of the downloaded contents of each buyer will be required. This preserves the*

buyers' privacy with respect to the merchant. It is even possible to establish some pre-payment protocol between buyer, transaction monitor and merchant so that the buyer account is charged after each content transfer without disclosing specific contents to the merchant. Another alternative is to protect the access to the P2P platform by means of some subscription account. In any case, payments do not need to be distributed; they can be centralized and simple protocols can be used for them without disclosing which specific contents are being transferred to buyers.

3.4 Tracing illegal redistributors

We now show that the proposed fingerprinting method allows identification of illegal redistributors of fingerprinted contents. Here, we distinguish between the basic protocol and the collusion-resistant version of the scheme.

3.4.1 Basic tracing protocol

Assuming that the embedding scheme is secure and robust enough so that malicious users cannot easily erase their fingerprints without making the content unusable (this is the standard marking assumption [9]), the following method can be used by a tracing authority to identify the source of an illegally redistributed copy.

Protocol 5 (Tracing)

1. *The fingerprint f of the illegally redistributed content X_f is extracted by the tracing authority using the extraction method and the secret extraction key K^e provided by the merchant.*

2. *The initial test set T_0 is built with the M buyers of the seed versions of the file.*

3. *Let $i := 0$.*

4. *The tracing authority contacts the buyers in the current set T_i. It also retrieves the hashes of the fingerprints of these buyers from the transaction monitor. This step requires the private key K_j^s of one parent of these buyers (the merchant in case of $i = 0$ and the selected ancestor in the set T_{i-1} otherwise) and the private key K_c^s of the transaction monitor. The fingerprints of the buyers of T_i are extracted using the extraction function and the secret extraction key K^e. The hash function h is then applied for each segment to obtain the fingerprints' hashes for all tested buyers. If any of the buyers' fingerprints produces a hash which does not match the corresponding record in the transaction monitor, the associated buyer will be accused of forgery (contract breach).*

5. *In case that no forgery occurs, the correlation test is performed with the fingerprints of the buyers in the current set T_i. This step is carried out as a simple bitstream correlation. Given the fingerprint f to be traced and the test fingerprint f' extracted from the copy $X_{f'}$ held by a buyer in T_i, both fingerprints with*

length L, the correlation $C(f, f')$ between f and f' can be computed as:

$$C(f, f') = \frac{1}{L} \sum_{j=1}^{L} (-1)^{f_j \oplus f'_j}, \tag{3.1}$$

where f_j and f'_j are, respectively, the j-th bits of f and f', and $\oplus$ refers to the exclusive-or operation. In case of forgery, this step can be computed with the hashes of the fingerprints instead of the fingerprints themselves. If the correlation of the hashes is equal to 1, the corresponding buyer is charged of unlawful distribution and the tracing protocol halts.

6. *If no buyer has been accused of illegal redistribution so far, there may be three outcomes of the previous step:*

   (a) *One or more buyers in T_i refuse to collaborate with the tracing authority in computing their correlations with f; in this case, depending on the correlation between the hash h_f and the hash(es) of the refusing buyer(s) (recorded in the transaction monitor), the refusing buyer(s) is(are) accused either of redistribution (if hashes are identical) or contract breach (otherwise). If the correlation between hashes is lower than 1, this correlation can be used as a replacement of the correlation between the fingerprints.*

   (b) *One buyer in T_i has $C(f, f') = 1$; in this case, this buyer is accused of the redistribution.*

   (c) *Otherwise, the buyer in T_i who has the maximum correlation with f is taken as the most likely ancestor of the buyer of the illegally redistributed copy; in this case, a new set T_{i+1} of buyers is built with all the children of this ancestor buyer, excluding any children buyers who have been already analyzed (remember that a buyer can have several parents). These children can be obtained from the transaction monitor (transaction records). Once the new set T_{i+1} is available, set $i := i + 1$ and go to Step 4.*

Although the maximum correlation criterion will be right most of the time, it cannot be discarded that a higher correlation might accidentally be obtained for a non-ancestor of the buyer of the illegally redistributed copy. For example, a descendant A of the illegal redistributor B may have as another ancestor a node C of the graph which is also ancestor of B. This would produce a high correlation with A but the chain from C to A skips the illegal redistributor B. In this situation, *backtracking* is required in the tracing protocol described above. A complete subnetwork would be analyzed until all nodes of the subgraph having no children are considered. When a complete subnetwork is exhausted, the element of T_i with the second maximum correlation would be chosen as the candidate ancestor of the illegal redistributor. When all elements of T_i have been considered without success (*i.e.* without being able to accuse anyone), the procedure would backtrack to the set T_{i-1}. Backtracking has been needed in a very small number of the simulations presented in Section 3.6.

To compute the correlation with a buyer's fingerprint and the traced fingerprint, the analyzed buyer must provide her copy of the content. A buyer may argue that she

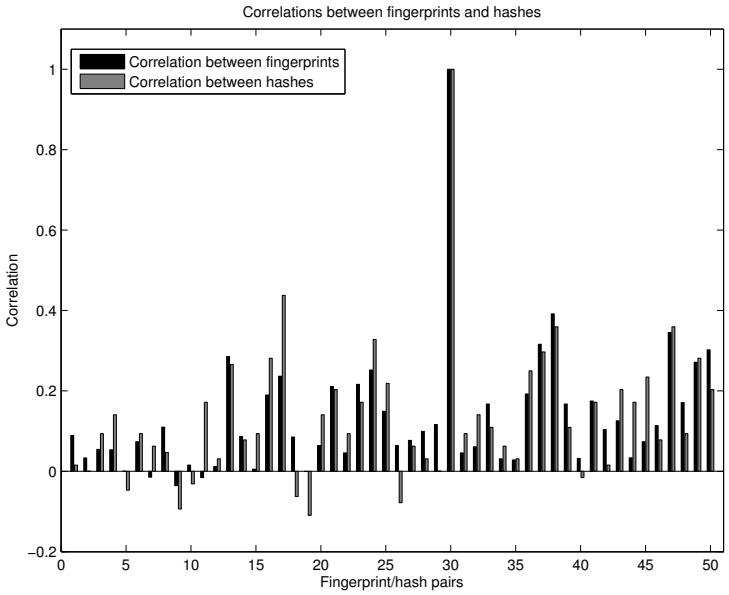
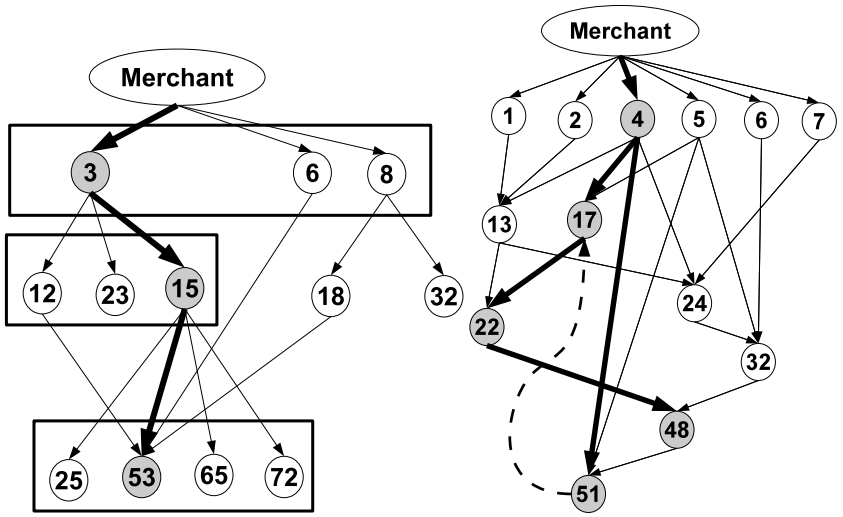


Figure 3.3: Correlations of fingerprints versus correlations of hashes

has lost or accidentally deleted the content file to refuse taking part in the test. Such a possibility should be limited in the contract of buyers for using the P2P distribution platform; yet, the lack of buyer co-operation can be circumvented as follows. Even if the actual content is not available, the hash of the fingerprint is stored at the transaction monitor. As a side effect of the way fingerprints and hashes are created, the bit correlation between hashes is a good estimate of the bit correlation between fingerprints. When two segments of two fingerprints are identical, they contribute to the overall correlation with a positive value. In this case, the bit hashes are also identical, and a similar positive effect occurs when the correlation is computed with hashes. For non-equal segments, the rest of the fingerprint contributes to the correlation with a value around 0 (on average), and the same goes for the hash bits of these segments (on average half of these bits would be equal and the other half would be different). Hence, the bit correlation between hashes can be used as a substitute for the bit correlation between fingerprints, which allows continuing the tracing process. This is illustrated in Figure 3.3, where fifty random pairs of fingerprints (with 128 segments and 32 bits per segment) and their corresponding hashes (128-bit long) of a simulated distribution graph have been used. The figure shows the correlations obtained with fingerprints (black bars) and hashes (grey bars). It can be seen that the results obtained with hashes and fingerprints are similar, but not identical. Hence, hashes should only be used as a last resort, since the errors in the correlation values could degrade the search. In any case, hashes can be used for a few cases during a search.

It may be argued that two different buyers may have the same hash for their fingerprints (hash collision). If the hashes have a large enough set of values, the probability



(a) Typical situation (b) Backtracking example

Figure 3.4: Tracing example showing a subgraph of the P2P content distribution scheme

of this collision can be low. If, for example, 40-bit long hashes were taken, there would be 2^{40} different hash values, whereas the population of the Earth is below 2^{33} people. Collisions of hash values would be very unlikely in that situation (though the use of anti-collusion codes would make it difficult to obtain that many different codewords).

Furthermore, it must be remarked that each buyer will have at least two parents in the proposed approach. Hence, even if one ascendant fails to provide her fingerprint for computing the correlation, the search will finally succeed by exploring a different branch of the distribution graph (backtracking).

A graphical representation of the tracing protocol is given in Figure 3.4. This figure provides two examples that are explained below. The problem of backtracking is that the fraction of the graph to be explored in a tracing case cannot be predicted or bounded *a priori*. In a worst-case (and non-realistic) situation, the whole set of buyers would have to be checked in order to identify the illegal redistributor. Although this very worst case has not appeared in any of the simulation experiments carried out for this paper (Section 3.6), it may be argued that there is no guarantee that such an extreme case will not occur. In order to bound the longest search to locate an unlawful distributor, the following system is proposed. The seed copies, and all copies obtained from their segments, must have an expiration time or counter after which all these fingerprinted copies will be removed from the P2P distribution system. For example, this expiration system would remove the copies from the download base (but not from the clients themselves, who would be able to preserve their purchased copies) if the download counter achieves some maximum value (*e.g.* 20,000 downloads). After that, the seed copies will be reset with new segments, and the distribution system will start from

scratch. The P2P software will keep a list of downloadable items and a list of expired items. In this way, the maximum theoretical number of tests per tracing will be limited to the expiration number (20,000 in the given example). In practice, as shown in Section 3.6, the cases requiring backtracking are not many and the actual number of tests will affect only a small fraction of the buyers in the same "partition" of the illegal redistributor: a simple search of the first segment of the fingerprint of the detected illegally redistributed copy can be used to determine which partition of the set of buyers needs to be examined.

We now give some examples of the operation of Protocol 5. Figure 3.4(a) shows an example of a P2P distribution network, more specifically a subnetwork of the full system represented by a directed graph from content sources (parents) to content destinations (children). The figure illustrates how Protocol 5 discovers that the fingerprint in the content (traced fingerprint) is the one of buyer B_{53}. The system begins testing the correlation between this traced fingerprint and a set T formed by all the children of the merchant. If $M = 10$, then $T_0 = \{B_1, B_2, \ldots, B_{10}\}$. In this case, three buyers among those in T_0, namely B_3, B_6 and B_8 have fingerprints with the top three correlations with the traced fingerprint (no wonder if one knows that B_{53} is the traced buyer, because B_3, B_6 and B_8 are ancestors of B_{53}). It turns out that B_3 is the one with the highest correlation (again, no surprise if one knows that B_{53} is the traced buyer; B_3 and B_6 are the most likely to have fingerprints with the highest correlation with B_{53}, since two of the parents of B_{53} are children of B_3 and B_6 is also a parent of B_{53}). The next iteration is performed with all the children of B_3, namely $T_1 = \{B_{12}, B_{15}, B_{23}\}$. The test yields the highest correlation with B_{15} (if one knows that B_{53} is the traced buyer, since B_{53} has four parents including B_{12} and B_{15}, the correlation of the fingerprints of the latter two buyers with the fingerprint of B_{53} must be around 0.25). Now, the set T_2 is formed with buyer B_{15}'s children: $T_2 = \{B_{25}, B_{53}, B_{65}, B_{72}\}$. In this situation, B_{53} will be found to have a fingerprint with correlation 1 with the traced fingerprint unless she refuses to take the correlation test. In any case, she will be accused of illegal redistribution, since a perfect match exists between the recorded fingerprint's hash for B_{53} and the hash of the traced fingerprint. In Figure 3.4(a), the nodes highlighted in grey are the ones yielding the highest correlations and rectangles are used to enclose the nodes that are involved in correlation tests. Note that only seven non-seed nodes are involved in correlation tests. With an average of three children per node, the network may easily include more than 100 nodes in two generations, meaning that 7% or less of the nodes would participate in those tests. More specific results about this issue are given in the simulated experiments presented in Section 3.6.

Figure 3.4(b) shows an example of a situation which requires backtracking, where B_{48} is the illegal redistributor of the content. The curved dotted arrow in the figure does not represent an edge of the graph, but the backtracking process. In this situation, the set $T_0 = \{B_1, B_2, \ldots, B_{10}\}$ is formed as in the previous example and the maximum correlation is obtained for B_4. Note that B_4 is an ancestor of B_{48} (as expected) but it shares a common child (B_{51}) with the illegal redistributor. The new set of candidates is constructed with B_4's children as $T_1 = \{B_{13}, B_{17}, B_{24}, B_{51}\}$. In this case, B_{51} is very likely to produce a very high correlation with the traced fingerprint (the one of B_{48}), because B_{48} is a parent of B_{51} and the other parent (B_4) is an ancestor of B_{48}. Once B_{51} is selected, her children (if any) and all her subgraph of descendants

would be examined without finding a correlation $C = 1$. Finally, after analyzing all the subgraph of descendants, backtracking occurs, hence going back to the set T_1 and picking the second highest correlation in the set, which is found for B_{17}, who is a true ancestor of the illegal redistributor (B_{48}). After that, two more iterations are required to find the illegal redistributor (descendants of B_{17} and descendants of B_{22}).

3.4.2 Collusion of malicious buyers

Fingerprinting schemes must provide some degree of collusion resistance in order to be able to trace forged copies created by advanced attackers. In this section, we show how the existing anti-collusion fingerprinting codes can be used also in the proposed distribution scheme. Hence, our scheme can be made as resistant against collusion as any of the existing anti-collusion techniques of the literature.

Under the usual marking assumption, error correcting codes are a typical solution to detect collusions [21]. Other approaches are based on more recent techniques, such as Tardos codes [42] or even newer codes based on them, like [36]. In the latter case, the marking assumption can be relaxed to a δ-marking assumption [36].

A special type of collusion that may occur is when a buyer tries to obtain different copies of the same content from the system to build a self-colluding copy and remove the fingerprint. To avoid this kind of attack, buyers will only be allowed to purchase one copy of the content through the P2P distribution software. Note that, even if the user stays pseudonymous versus the transaction monitor, her pseudonym is a stable one, so the transaction monitor can prevent the user from buying the same content twice. In case a buyer needs to purchase the content again (due to accidental removal, hardware wreckage or any other unwanted situation), she will have to use a standard centralized purchasing system. The P2P solution can only be used once. Since not many content losses are expected, this does not represent a serious disadvantage for the proposed system.

We describe how to add collusion resistance to our scheme:

- Each segment is encoded with an anti-collusion code which can be used to reconstruct the segment of one of the colluders. Since the merchant embeds the fingerprints of the seed buyers into the content, an honest merchant suffices to guarantee that all the segments are encoded using this specific codebook. In this way, if a set of colluders fabricate a copy of the content and redistribute it over the Internet, each segment can be decoded to recover the corresponding segment of one of the colluders.

- The fingerprints must be constructed in such a way that their hashes are also codewords of some collusion-resistant code. In this way, after a collusion, when the segments have already been reconstructed, the hash of at least one of the colluders will be obtained. In this case, the proxies will be responsible for constructing a valid codeword for each hash, with the appropriate structure. For example, for error correcting codes, the "data" bits of the hash can be chosen randomly, whereas specific parents having the required hash bit will be picked up for the redundancy bits of the hash. The proxy can contact parents subsequently, by requiring the specific hash bit for a given segment, and only those

having the specific hash for that segment would accept becoming the source for that specific fragment of the content.

Collusion resistance is thus obtained by a 2-layer collusion-resistant coding of the fingerprints:

- The anti-collusion code used for the segments of the fingerprint (segment-level code).
- The anti-collusion code used for the hash of the fingerprint (hash-level code).

Fortunately, for the segment-level code, the number of codewords does not need to be very high, since we only need a number of different codewords equal to the number of seed buyers (M), and this number will always be small (*e.g.* $M = 10$ is used in the experiments presented below). In the case of $M = 10$ and four colluders, Tardos-like codes with codewords around 100 bit long (or less) would be possible according to the results of [36].

The hash-level code must be designed for a larger set of users (for example $N =$ 20,000 if the graph is reset after 20,000 transactions as suggested above). In this way, the fingerprint size (in bits) would be equal to 100 (the segment size) multiplied by the longer size of the hash-level code used for the hashes of the fingerprints. This means that the length of the fingerprint would be of the same order as the most efficient fingerprinting code that could be found, multiplied by a constant (the length of the segment-level anti-collusion code used for the fingerprints' segments). There is a penalty for using the two-layer fingerprint encoding, but it does not square the length of a Tardos-like code as one might think, since the length of the segment-level code can be kept relatively small (it should work only for $M \ll N$ different users).

We suggest that seed buyers be dummy buyers created by the merchant, rather than real buyers of the content. With our suggestion, seed buyers will not participate in any collusion, so their fingerprints do not need to satisfy the above condition that their hashes be codewords of a collusion-resistant code. Hence, the merchant can enforce that, for each segment, an equal number of seed buyers have a '1' and a '0' hash bit. This maximizes the chances that a proxy can find parents with the appropriate hash bit for a specific position of the hash of a real buyer's fingerprint.

We remark that the above solution has exactly the same problems as standard collusion-resistant fingerprinting techniques, mostly related to the lengths required for the codewords in practical situations. In any case, [36] provides short enough codes to be used in a practical implementation of this proposal.

The following procedure is run to trace an illegal redistributor after collusion:

1. The segments of the colluders are reconstructed using the appropriate anti-collusion code.

   There are at least two ways for buyers to collude, namely, bit collusion and segment collusion. In the first case, buyers do not know the structure of the fingerprint (for example if such structure is determined by a secret key). Colluders just look for differing bits in their copies of the content and set those bits randomly in the forged copy. This causes the segment structure to be disturbed,

with new segments appearing that not only were not present in the seed buyers' fingerprints, but are not even valid codewords of the segment-level anti-collusion code. Using the anti-collusion code, each segment can be decoded to match the corresponding segment of one of the colluders. In the second case (segment collusion), the traitors know about the fingerprint structure and create a new fingerprint with valid segments, by picking segments randomly among those of the set of colluders' fingerprints. Now, the segments will be valid codewords of the corresponding segment-level anti-collusion code, but the hash of the fingerprint will not be a valid codeword of the hash-level anti-collusion code. To be able to produce a valid hash-level codeword, many colluders would be needed (to have enough options for all bits of the hash). This kind of collusion is likely to require more colluders than the maximum size c of the collusions resisted by the anti-collusion code itself. For example, if the anti-collusion code can withstand collusions of size up to $c = 5$ and 9 or more colluders are required to produce a valid hash-level codeword, the real limit is the anti-collusion capacity of the code (5 in the example): if 5 or more buyers can defeat the anti-collusion code, there is no need to produce a valid hash-level codeword involving 9 buyers.

2. The hash of the fingerprint must be reconstructed.

   The hash function can be applied to each reconstructed segment and, after that, the anti-collusion code used for the hash shall be used to obtain the hash of one of the colluders.

3. The basic tracing protocol introduced in Section 3.4.1 must be modified and an advanced version will be used to treat collusion.

   The exit condition of the protocol cannot be to find a correlation $C(f, f') = 1$, because the reconstructed fingerprint does not contain segments from a single fingerprint, but possibly a mixture of the segments of the colluders' fingerprints. After finding the maximum correlation in each set of analyzed buyers (*e.g.* the seed buyers), the children of the corresponding buyer are considered as the candidate set of nodes to explore. Prior to contacting this set of buyers to compute their correlation, the hash of these children will be recovered from the transaction monitor. If the hash of any of these buyers is identical to the reconstructed hash, then the corresponding buyer will be considered as the malicious buyer involved in the collusion. Note that it is enough to have one parent of each buyer to decrypt the hash stored at the transaction monitor. This means that the transaction monitor only needs the private key of one parent to decode the hashes of all her children and no other party is required in this step.

A proof of concept of this idea using dual Hamming error correcting codes is also provided in Section 3.6.

3.5 Security analysis

In this section, we first specify the security assumptions of our scheme. We then analyze to what extent buyers can preserve privacy, *i.e.* to what extent it needs to become

known that a certain buyer has bought a specific piece of content and to what extent the specific fingerprinted copy held by a buyer remains only known to that buyer. We finally examine buyer frameproofness vs a malicious merchant.

3.5.1 Security assumptions

In the proposed scheme, the proxies and the transaction monitor do not know real identities, only pseudonyms (usernames). Hence, neither the proxies nor the transaction monitor can break the privacy of buyers by themselves. In what regards privacy, the fact that a given buyer has purchased some specific content can only be leaked if the merchant and at least one of the proxies or the transaction monitor are malicious. The merchant is the only party having access to the real identities.

The only threat to buyer security (resulting in an innocent buyer being framed) is a coalition of all proxies chosen by a buyer. Proxies have access to the cleartext of the content's fragments (since they have access to session keys). In addition (Protocol 4, Step 5), proxies need to exchange the fragments of the child buyers' fingerprint hash, meaning that proxies need to have contact between them during the process. If all the proxies chosen by a buyer collude, they can replicate the content transferred to the buyer by joining the different pieces together and re-distributing the content illegally to frame an innocent buyer. This paper assumes that proxies are honest, and a detailed analysis of malicious proxies is left for the future research.

The tracing algorithm requires that at least one of the parents of a buyer provide her secret key to obtain the fingerprint's hash stored at the transaction monitor. If a buyer refuses to co-operate by providing her fingerprint's correlation with the traced fingerprint, it would be required that at least one of the parents of the buyer decrypt her child's fingerprint hash. If all the parents of a non co-operative child refused to do so, the system would not be able to trace the child if she were the illegal redistributor. However, it must be taken into account that parents and children are anonymous to each other due to the use of Protocol 4. Hence, parents do no have any rational reason to cheat the system to favor an unknown child. In addition, cheating parents would have to pay some punishment (fine) due to contract breach.

Parents are also expected to provide the fingerprint's hash bit of each fragment. They could cheat and change the bit, but, again, doing this would favor an unknown child. A simple solution consists in having the fingerprint's hash bits signed by the merchant in origin (the signature can include the fingerprint's hash bit plus a standard hash of the fragment). In this way, parents would not be able to alter the fingerprint's hash bit of each fragment, since this would require having access to the merchant's private key.

3.5.2 Buyer privacy

Buyer privacy in our scheme is inversely proportional to the size of the fraction of buyers affected by the correlation tests carried out by Protocol 5 when tracing illegal redistributors. Indeed, testing correlation forces the tested buyer to reveal her fingerprinted copy and hence to lose her privacy. Hence, we will focus on the fraction of tested buyers.

Table 3.1: Maximum number of correlation tests for buyers of different generations assuming that all buyers in the same generation have the same number of children and no backtracking is needed

Gen.	# Buyers	Maximum expected correlation tests per buyer
1	M	M
2	M	$M + n(k-1)$
3	$2M$	$M + n(k-1) + n(k-2)$
4	$4M$	$M + n(k-1) + n(k-2) + n(k-3)$
$\vdots$	$\vdots$	$\vdots$
j	$2^{j-2}M$	$M + n(k-1) + n(k-2) + \cdots + n(k-j+1)$
$\vdots$	$\vdots$	$\vdots$
k	$2^{k-2}M$	$M + n(k-1) + n(k-2) + \cdots + n$

The average number of correlation tests in the course of a redistribution investigation depends on the structure and size of the graph. However, some expressions for this number can be derived if the following assumptions are made:

1. The first generation is formed by the M seed buyers.

2. At each generation, the population increases by 100%. This means that, on average, each P2P buyer sends the whole content allowing to satisfy a new buyer (a new copy of the entire content). Hence, the second generation would be formed by M new buyers. The third generation would be formed by $2M$ buyers, and so on. With this assumption, the population increases exponentially after each generation. For example, after six generations, the population would be $M + M + 2M + 4M + 8M + 16M = 32M$. If k is the number of generations, the total population is $N = 2^{k-1}M$.

3. Let n be average number of parents per buyer. If all possible parents have the same probability of being chosen, after k generations the buyers of the first generation will have $n(k-1)$ children on average; the buyers of the second generation will have $n(k-2)$ children on average and, in general, the number of children per buyer will be $n(k-j)$ for the j-th generation. This makes it possible to estimate the expected value of the maximum number of correlation tests required to locate a particular buyer at each generation (if no backtracking occurs).

4. The maximum number of correlation tests per generation without backtracking is shown in Table 3.1, where it is assumed that all buyers within the same generation have the same number of children. For example, for buyers of the third generation, the worst case is when we need to explore all buyers in the first generation (M), the children of one of them ($n(k-1)$) and the children of the chosen child ($n(k-2)$). Hence $M + n(k-1) + n(k-2)$ is the maximum required number of correlation tests in case we need two iterations of the tracing protocol.

Note that the "worst-case" figures in Table 3.1 are only valid if all buyers in the same generation have the same number of children. If some buyers have more children,

more correlation tests will be required for them, which will increase the number of tests and exceed the corresponding figure in Table 3.1. For example, consider $M = 10$, $n = 3$ and two generations ($k = 2$). Assume buyers B_1, B_2 and B_3 have seven common children, namely, B_{11}, B_{12}, ..., B_{17} (each of these seven children has B_1, B_2 and B_3 as parents). Tracing any of those second-generation children will require seven correlation tests plus the $M = 10$ tests for the first generation, which always takes $M = 10$ tests. Even if the remaining three buyers B_{18}, B_{19} and B_{20} in the second generation only require one correlation test each, the average number of tests for the second generation of buyers will be $M + (7 \cdot 7 + 3 \cdot 1)/10 = M + 5.2$. This is more than the value $M + n(k-1) = M + 3 \cdot 1 = M + 3$ shown in Table 3.1 for the second generation.

In addition, it must be taken into account that backtracking has not been considered in Table 3.1; however, this is a quite realistic approach since simulations show that only a small fraction of traced buyers require backtracking.

Lemma 2 *If k is the number of generations and all buyers in the same generation have the same number of children, the expected value of the maximum number of correlation tests is*

$$A = M + 2^{1-k}(k+1)n - \frac{1}{2}(4 + k - k^2)n.$$

Proof Firstly, the sum in the j-th row of Table 3.1 can be simplified as follows

$$M + n\left((j-1)k - \frac{j(j-1)}{2}\right).$$

Now, compute the weighted average of Table 3.1 taking as weights the fraction of buyers in each row

$$A = M + n\frac{\displaystyle\sum_{j=2}^{k} 2^{j-2}\left((j-1)k - \frac{j(j-1)}{2}\right)}{2^{k-1}},$$

from which

$$A = M + n\frac{\left(k - \frac{2^{k+1}\left(4+k-k^2\right)}{8} + 1\right)}{2^{k-1}}.$$

The expression of the lemma follows. □

Hence, with the assumptions of Lemma 2, the expected maximum number of correlation tests grows quadratically with k. Since $k = \lceil \log_2(N/M) \rceil$, with these assumptions, the expected search complexity (number of correlation tests) is quadratic logarithmic in the population size (total number of buyers). For example, for $M = 10$ (10 seed buyers), $n = 3$ (three parents per node on average) and $k = 4$ generations, the expected maximum number of tests would be $A = 23.875$.

Lemma 3 *Under the same assumptions of Lemma 2, the expected value of the maximum number of correlation tests excluding the first generation and not counting the M seed buyers (who must always be examined) is*

$$A' = \frac{2^{1-k}(k+1)n - \frac{1}{2}\left(4+k-k^2\right)n}{1-2^{1-k}}.$$

Proof The expression follows by subtracting M in the values of Table 3.1 and computing the weighted average excluding the M seed buyers. □

Corollary 1 *Under the same assumptions of Lemma 2, the expected maximum fraction R of non-seed buyers affected by a correlation test is*

$$R = \frac{2^{1-k}(k+1) - \frac{1}{2}\left(4+k-k^2\right)}{\left(1-2^{1-k}\right)\left(2^{k-1}-1\right)}\frac{n}{M}. \tag{3.2}$$

The above fraction decreases asymptotically towards 0 as k grows, that is, as the population grows, the fraction of non-seed buyers who must surrender their privacy in a correlation test decreases exponentially.

Proof The expected maximum fraction is $A'/(N-M)$, where A' is given by Lemma 3 and $N = 2^{k-1}M$. The numerator grows quadratically with k, whereas the denominator grows exponentially with k. The corollary follows. □

Figure 3.5 depicts R (Expression 3.2) for $M = 10$, $n = 3$ and $k = 2, 3, \ldots, 20$. It can be seen that for $k = 3$ (3 generations) an expected maximum of 26.7% non-seed nodes need to be tested when no backtracking occurs; for $k = 4$ it is 22.7%; for $k = 7$, it is 9.3%; for $k = 12$ it is 0.9%, etc. As the population grows, the tests will affect only a very small percentage of the buyers, while the rest will remain completely undisturbed and private.

3.5.3 Buyer frameproofness

Buyer frameproofness relates to the case of a malicious merchant trying to frame an honest buyer by accusing her of being the source of an illegally redistributed content.

In the proposed system, the merchant either does not have access to any fingerprinted copy of the content (if a secure multiparty computation scheme is used to create the seed copies) or he has access only to the fingerprints of the M seed buyers:

- In the first case, there is no way for the merchant to produce the fingerprint of any particular buyer (random guess is not an option if fingerprints are long enough) and, therefore, all buyers are protected from the merchant's malicious behavior.
- In the second case, the merchant might use the fingerprints of the M seed buyers to frame them, by falsely accusing them of redistribution or collusion (the merchant might create a false colluded copy using the fingerprints of the seed buyers). To avert such a dishonest behavior, the seed buyers should receive a guarantee that the merchant is not going to use their fingerprinted copies to

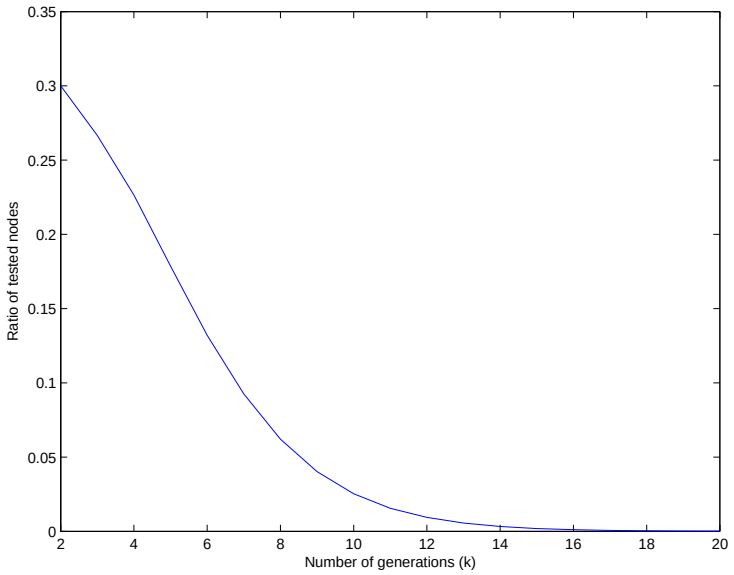


Figure 3.5: Expected fraction R of non-seed buyers involved in correlation tests, for $M = 10$, $n = 3$ and $k = 2, 3, \cdots, 20$

frame them (this does not leave an honest merchant helpless, though, because he will indeed be able to detect and possibly blacklist any really colluding seed buyers). A simpler and perhaps better alternative is *for the M seed buyers not to be real buyers, but dummy buyers created by the merchant to bootstrap the P2P distribution protocol*; the first real buyer is the $M + 1$-th one. Even if the seed buyers are protected against false redistribution and collusion charges, the merchant could still try to produce a combination of the seed copies with the hope that it would have a high correlation with some descendant of the seed buyers who could then be falsely accused. This possibility can be neglected. For example, if 10 values are possible for each segment, and 128 segments exist, there would be 10^{128} different possible fingerprints. The probability to build a correct fingerprint even if every person of the Earth is a buyer in this system is infinitesimal. The probability to build an existing hash to frame an innocent buyer with collusion charges would not be that small, but still negligible if the set of hash values is large enough.

It is worth pointing out that the merchant does not need to have access to the extracted fingerprints in the tracing protocol. As detailed in Protocol 5, it is an independent (trustworthy) tracing authority who needs the correlations between fingerprints to proceed with the search. If the merchant does not have access to fingerprints (not even in the course of a tracing investigation) she will not be able to embed a true fingerprint in the content to make a false accusation on an honest buyer in a future investigation.

3.6 Simulation results

This section presents a set of simulated experiments to illustrate the properties of the proposed system: buyer privacy, robustness against non-collaborative buyers and collusion resistance.

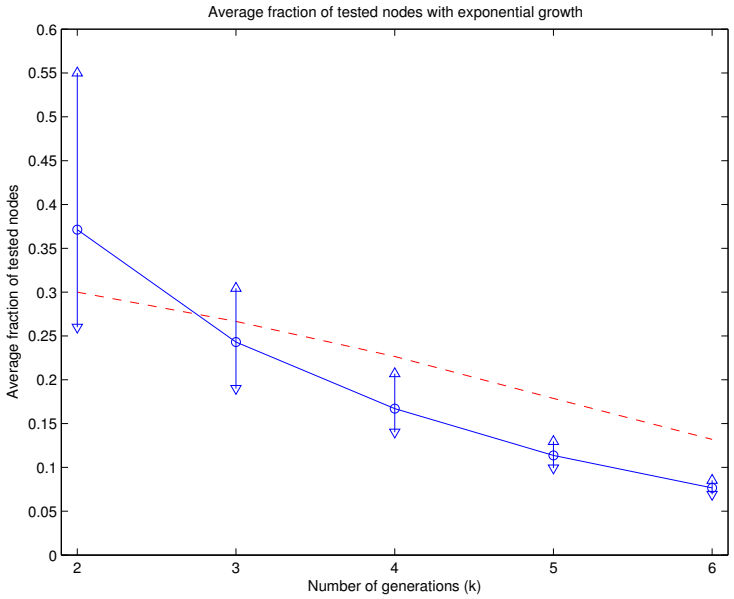


Figure 3.6: Average fraction of non-seed buyers affected by correlation tests in an exponentially growing population: simulation results (solid) and theoretical expected maximum value R according to Corollary 1 (dashed). Vertical solid lines are max-min intervals.

3.6.1 Buyer privacy

In all simulations presented in this section, the recombined fingerprints were 4096-bit sequences divided into 128 segments of 32 bits each. The first simulation to confirm the results presented in Section 3.5.2 consisted of producing different generations of buyers using an exponential growth approach and checking the average number of correlation tests required to identify the buyers. The number of seed buyers was taken to be $M = 10$ and each buyer could have between two and four parents which were chosen at random from all the previous generations (not only the immediately previous one). This means that the average number of parents per non-seed nodes was $n = 3$. The simulations shown in Table 3.2 were carried out, and a comparison of the average number of correlation tests with the expected fraction introduced in Section 3.5.2 is shown in Figure 3.6.

The results in Table 3.2 show a single simulation and the average of 100 simulations

Table 3.2: Average number and percentage of correlation tests on non-seed buyers in an exponentially growing population

Generation	Population	Average correlation tests		Backtracking (100 sim.)
		1 simulation	100 simulations	
$k = 2$	$N = 20$	3.40 (34.0%)	3.71 (37.1%)	0%
$k = 3$	$N = 40$	6.93 (23.1%)	7.29 (24.3%)	0%
$k = 4$	$N = 80$	12.26 (17.5%)	11.69 (16.7%)	0.6%
$k = 5$	$N = 160$	18.99 (12.7%)	17.05 (11.4%)	1.2%
$k = 6$	$N = 320$	24.31 (7.8%)	23.76 (7.7%)	2.7%

Table 3.3: Average number and percentage of correlation tests on non-seed buyers in a linearly growing population

Generations	Population	Average correlation tests		Backtracking (100 sim.)
		1 simulation	100 simulations	
$k = 2$	$N = 20$	3.40 (34.0%)	3.71 (37.1%)	0%
$k = 3$	$N = 30$	5.40 (27.0%)	5.54 (27.7%)	0%
$k = 4$	$N = 40$	6.20 (20.7%)	6.76 (22.5%)	0.17%
$k = 5$	$N = 50$	7.45 (18.6%)	8.15 (20.4%)	0.45%
$k = 6$	$N = 60$	8.20 (16.4%)	8.95 (17.9%)	0.42%

Table 3.4: Average number and percentage of correlation tests on non-seed buyers: comparison between exponential and linear growth for the same population

Population	Exponential growth		Linear growth	
	Generations	Average tests (100 simul.)	Generations	Average tests (100 simul.)
$N = 20$	$k = 2$	3.71 (37.1%)	$k = 2$	3.71 (37.1%)
$N = 40$	$k = 3$	7.29 (24.3%)	$k = 4$	6.90 (23.0%)
$N = 80$	$k = 4$	11.69 (16.7%)	$k = 8$	10.62 (15.2%)
$N = 160$	$k = 5$	17.05 (11.4%)	$k = 16$	15.43 (10.3%)
$N = 320$	$k = 6$	23.76 (7.7%)	$k = 32$	22.23 (7.2%)

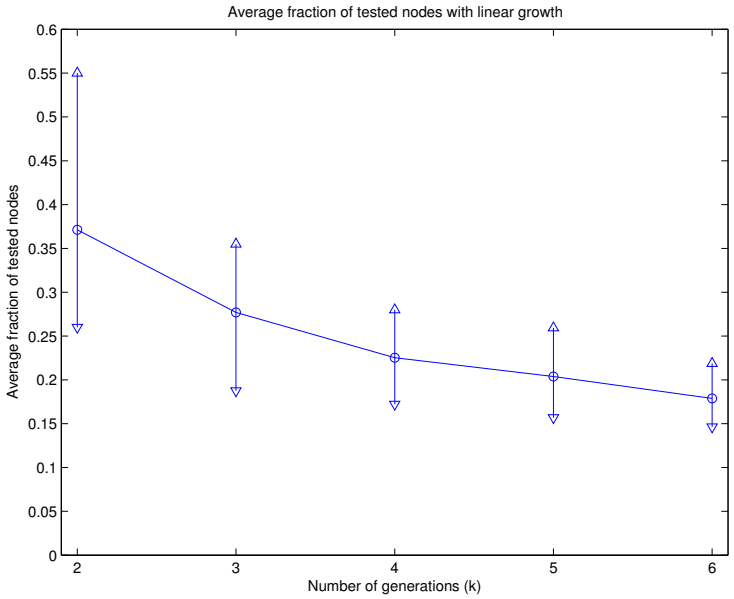


Figure 3.7: Average fraction of non-seed buyers affected by correlation tests in a linearly growing population (simulation results). Vertical solid lines are max-min intervals.

with 100 different seeds in the pseudo-random number generator in order to reduce the bias of the results. It can be seen that no significant differences appeared between 1 and 100 simulations. The last column represents the average percentage of buyers requiring backtracking in the 100 simulations. Not surprisingly, as the network (graph) became larger, more buyers required backtracking, but the percentage was always small.

Figure 3.6 shows intervals for the average fraction of non-seed buyers affected by correlation tests as the number of generations grew. For each number of generations, the corresponding vertical solid line represents an interval with the up triangle showing the maximum fraction in 100 simulations, the down triangle showing the minimum fraction and the circle showing the average fraction; these average fractions correspond to the percentages given in Table 3.2 for 100 simulations. As discussed in Section 3.5.2, the theoretical expected maximum fraction of tested non-seed buyers (dashed line) can be exceeded if the number of generations is small, due to the effect of some parents having more than the average number of children. This situation is compensated as more generations are produced and the simulated fraction goes below the theoretical value already for $k > 3$, although the interval for $k = 3$ shows that, for that number of generations, some simulations still yielded fractions above the theoretical value. In any case, as predicted in Section 3.5.2, the fraction of non-seed buyers affected by *one* correlation test decreased to zero as the number of generations grew: the more buyers involved, the higher the probability that a buyer did not need to surrender her privacy in one particular correlation test. However, as the population grows, the number of illegal

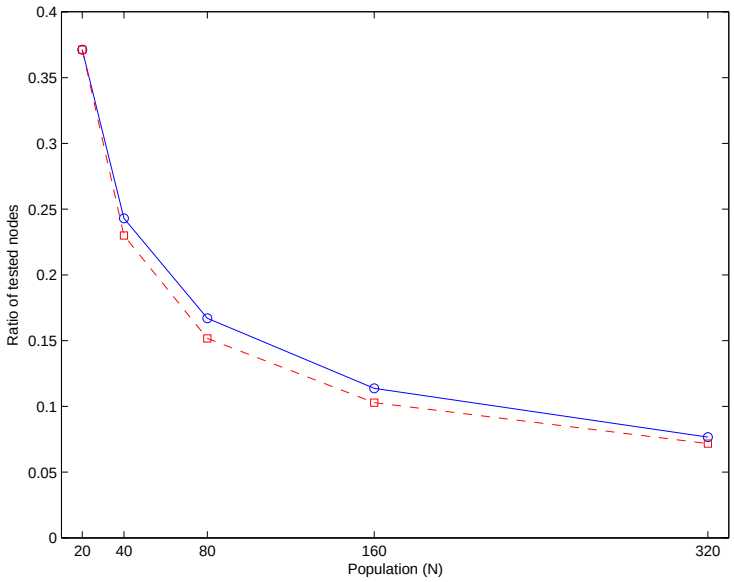


Figure 3.8: Average fraction of non-seed buyers affected by correlation tests: comparison between exponential growth (circle, solid) vs linear growth (square, dashed) for the same population. Abscissae is population size.

redistributions may also increase and more correlation tests may be needed to investigate them; as the number of required correlation tests increases, the probability that a non-seed buyer is affected by them (and therefore loses her privacy) also increases.

It may appear that the percentage of buyers involved in correlation tests in the course of an investigation decreases to zero because of the exponential increase in population occurring at each generation. However, this is not the case. The decrease of this ratio of tested buyers depends on the population and not on the particular way it grows. To illustrate this process, the following simulations were performed with a population growing linearly at each generation:

1. The first generation was, again, formed by the $M = 10$ seed buyers who obtain their fingerprinted contents from the merchant.

2. At each new generation, $M = 10$ new buyers obtained their contents from a variable number of parents between two and four (and thus, the average number of parents was, again, $n = 3$).

3. With this scenario, the population N increased linearly with the number of generations: there were $N = kM$ buyers after the k-th generation.

Table 3.3 illustrates this issue. It can be seen that the fraction of tested buyers decreased with the number of generations. In this case, the decrease was linear and not exponential, since the population increased linearly with k. This is also illustrated

Table 3.5: Ratio of tested nodes and number of backtrackings required for different probabilities of non-collaboration and numbers of generations

Non-collaboration probability	Generations	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$
	Population	20	40	80	160	320	640
0.0	Tests	34.0%	23.1%	17.5%	12.7%	7.8%	5.5%
	Backtracking	0	0	0	0	3	31
0.1	Tests	33.0%	26.7%	16.6%	11.2%	7.8%	5.7%
	Backtracking	0	0	1	4	23	56
0.2	Tests	39.0%	25.4%	17.0%	12.0%	7.6%	5.7%
	Backtracking	0	0	0	4	21	63
0.3	Tests	33.0%	23.2%	17.8%	11.3%	8.2%	6.8%
	Backtracking	0	0	0	5	22	102
0.4	Tests	40.0%	22.3%	16.6%	11.2%	9.4%	6.4%
	Backtracking	0	0	0	6	26	100
0.5	Tests	40.0%	24.7%	16.9%	12.2%	9.0%	8.4%
	Backtracking	0	0	1	5	30	139

in Figure 3.7 by means of interval plots. The seeds of the pseudo-random number generator were adjusted such that the results for two generations ($N = 20$) were the same as those presented in Table 3.2 for the exponential growth.

We present also simulation results comparing the linear and exponential growths scenarios *for the same population*. The results are shown in Table 3.4 and Figure 3.8. It can be seen that, when populations are of the same size, the results are almost identical irrespective of the number of generations and the growth model (exponential or linear). Again, the seeds of the pseudo-random number generator were adjusted so that the results for two generations ($N = 20$) were the same for both growth models.

3.6.2 Non-collaborative buyers

One of the conditions of the suggested protocols is that innocent buyers collaborate in the computation of correlations in order to trace an illegal redistributor. Of course, buyers will have to accept the license of the P2P distribution software and the terms of service which must state that non-collaborative buyers may be charged with contract breach and could be fined by the merchant (the transaction monitor can report the usernames of buyers who have refused to collaborate). Nevertheless, some buyers may still argue a *force majeure* situation which could have prevented them form collaborating even though they were willing to do so. For example, a buyer can argue a hardware wreckage, corrupted data, stolen devices, or some other plausible situation. In any of these cases, the graph search can still proceed using the correlations between fingerprints' hashes (which are stored in the transaction monitor) instead of the true fingerprints. Of course, the number of times a buyer can argue such kind of justified reason not to collaborate should be limited by the terms of service.

Since the correlation between hashes is only an approximation of the true correlation, this would produce some degradation in the search, possibly leading to more backtracking cases. This issue is analyzed in Table 3.5 where the column "Non-collaboration probability" refers to the probability that buyers do not collaborate in computing the correlation of fingerprints, so that correlations of hashes have to be used instead. Simulations have been performed for graphs with 2 to 7 generations, $M = 10$ seed buyers and an average number of parents $n = 3$ for each buyer. The non-collaboration probabilities range from 0 (all buyers collaborate) to 0.5 (on average, 50% buyers do no collaborate). The latter case would not be very realistic, since punishment would occur in case of non-collaboration. The results provided for probability 0.0 are exactly the same as those in Table 3.2 for one simulation. Table 3.5 provides two results, namely, the percentage of nodes taking part in correlation tests (without taking into account the seed buyers) and the number of cases requiring backtracking. The main differences can be appreciated when the graph reaches a relatively large size ($k = 6$ and $k = 7$). In those cases, it can be noticed that the number of searches requiring backtracking increases with the probability of non-collaboration, which results in an increased number of tested nodes. In addition, it can be observed that the degradation in the search is limited, and the ratio of tested nodes still decreases as the population grows even in the quite unrealistic case of having a 50% non-collaborative buyers.

3.6.3 Collusion resistance

In this section, experiments conducted with the anti-collusion version of the scheme suggested in Section 3.4.2 are presented. This simulation is a proof of concept. In a practical implementation, other codes and parameters should possibly be used. However, this implementation shows that the method suggested to fight collusion is more than a theoretical possibility.

The details of the implementation are as follows:

- A dual Hamming code $DH(31, 5)$ was used to encode the segments. $2^5 = 32$ values were thus possible for each segment. Each segment had 5 bits of data and 26 redundancy bits. This code can be used to detect collusions of two buyers.

- A dual Hamming code $DH(1023, 10)$, which also detects collusions of two buyers, was used to encode the hash of the fingerprint. Hence, $2^{10} = 1024$ different hashes existed, with 10 bits of data and 1013 bits of redundancy. This number of hashes would not be enough for a real implementation of the method, but it sufficed for this proof of concept.

- With these choices, the fingerprints were formed based on 1023 segments, each of which consisting of 31 bits. Hence, the fingerprints were $1023 \cdot 31 = 31{,}713$-bit long. The multimedia content had to be split into 1023 fragments, carrying each 31 embedded bits. Possibly, a better choice for a practical implementation with error-correcting codes would be Reed-Solomon (RS) codes instead of dual Hamming codes. In that case, the segments would represent symbols of the code (segment-level code) and the hash of the fingerprint could be an RS codeword

(hash-level code). Note that high-capacity robust watermarking schemes exist for embedding that amount of information. For example, the method proposed in [23] allows embedding up to 11,000 bits in a second of audio.

- 10 seed buyers were generated ($M = 10$). The hash of each segment was computed by simply selecting the third data bit of each gene. This is not a sophisticated hash and is obviously quite insecure, but it sufficed for simulation purposes. More advanced hashing techniques would be required in practice.

- For each gene, exactly five seed buyers had a '0' hash and the other five had a '1' hash. As pointed out in Section 3.4.2, this maximized the chances that a proxy could find parents with segments having the hash bit values required to build any hash-level anti-collusion codeword.

- An exponential increase of the population was assumed: 6 generations were created, resulting in a total population of $10 \cdot 2^5 = 320$ buyers (including the seed buyers).

- When non-seed buyers downloaded the content, the first 10 segments were chosen randomly between 2 and 4 parents. This yielded the 10 data bits of the hash of the fingerprint. The remaining 1013 bits of the hash had to be such that a codeword of the $DH(1023, 10)$ code was obtained. This was achieved by requesting fragments with segments that carry the appropriate hash bit to the current set of parents. If no parent with the required bit was found, the proxy looked for a new parent with an appropriate segment. This new parent was included in the set of parents of that buyer and was considered as a potential parent for the remaining fragments (segments and hash bits).

After generating a random population with these settings, the actual number of parents per (non-seed) buyer ranged from 4 to 11, with an average of $n = 9.09$ parents per buyer. Hence the privacy results could not be directly compared with those of the previous section (for which the average number of parents per buyer was around $n = 3$).

With these settings, 200 random bit collusions were generated. For each collusion, a new fingerprint was created by choosing randomly a new fingerprint's bit when the bits of the colluders differed. Hence, after the collusion, the obtained forged copy had a non-codeword embedded into it, both at the segment level and at the hash level. This is the standard marking assumption. For each forged copy, the advanced tracing system described in Section 3.4.2 was applied, by decoding the segments and the fingerprint's hash using the $DH(31, 5)$ and $DH(1023, 10)$ codes, respectively. Note that, with this approach, the colluders themselves did not need to participate in the search. When a buyer was selected as the most likely ancestor of the colluder, the hashes of her children were examined. If a match occurred for the hash of the fingerprint, the corresponding child buyer was the traitor. In case that hash collisions are allowed, some additional investigations would be required to guarantee that the selected buyer is a colluder, but this simplified scenario did not require further tests. After these 200 experiments, the average number of tests to find the colluder (with neither false positives nor false negatives) was 47.77 or 14.8% of non-seed buyers. This is much below the theoretical

maximum expected value, which can be obtained using Expression 3.2 for $M = 10$, $n = 9.09$ and $k = 6$ as $R = 0.400$ or 40.0% of non-seed nodes. Thus, even in case of collusion, the number of non-seed nodes involved in correlation tests decreases to zero as the population grows.

3.7 Conclusion

We have presented a recombination fingerprinting scheme designed for P2P content distribution. The proposed scheme allows the merchant to trace unlawful redistributors of the P2P distributed content. The merchant knows at most the fingerprinted copies of the seed buyers, but not the fingerprinted copies of non-seed buyers (the vast majority). Hence, the merchant does not know the identities of non-seed buyers. Whenever an illegal redistribution needs to be traced, only a small fraction of honest users must surrender their privacy by providing their fingerprinted copies (quasi-privacy). Our scheme also offers collusion resistance against dishonest buyers trying to create a forged copy without any of their fingerprints. Finally, a malicious merchant is most likely to fail in using the fingerprinted copies of seed buyers to try to frame an honest non-seed buyer (buyer frameproofness).

As mentioned above, future research will involve designing backtrack-free protocols to trace illegal redistributors, in such a way that the fraction of honest buyers losing their privacy in case of collusion tracing is further reduced. Using timestamps that can be retrieved from an illegally redistributed content seems a promising way to shorten the searches and avoid many cases of backtracking. An analysis of the vulnerability of the proposed scheme against malicious proxies, who may even collude with other parties (such as the merchant or the transaction monitor) is also left for the future research.

Chapter 4

Conclusions

The results presented in this Deliverable Report are included in WP3: Secure Electronic Commerce and Digital Content Distribution of the ARES Project and, more precisely, cover the following objective of this Work Package:

- *Design and implement asymmetric and anonymous fingerprinting systems: We will analyze the existing zero-knowledge fingerprinting proposals in order to obtain new solutions that could be efficiently implemented. With input from WP4 we will focus on new anonymous fingerprinting proposals in order to reconcile the identification of users that illegally redistribute copyrighted content with the privacy of honest users.*

This objective has been achieved and two novel anonymous fingerprinting proposals have been described in Chapters 2 [22] and 3 [32]. Both proposals satisfy the requirements but take two completely different approaches. On the one hand, the protocols of Chapter 2 provide a game theoretic solution whereby buyers cooperate in fingerprinting embedding and content distribution, and redistributor tracing is guaranteed by using registered fingerprints. On the other hand, the solution of Chapter 3 reduces the involvement of buyers for embedding since, in fact, embedding only occurs for a few seed buyers whereas recombination of their fingerprints suffices for creating different identifiers for subsequent buyers. In this case, redistributor tracing is achieved by means of a graph search using a binary correlation function to choose the most likely ancestor of the traced buyer. Some cooperation of buyers is required in the second solution during the search process (although the search can still succeed even if some or many buyers refuse to cooperate).

The protocols of Chapter 2 require buyers cooperation for embedding but not for tracing, unlike the system proposed in Chapter 3. In any case, it appears that any practical and scalable anonymous fingerprinting protocol using a P2P distribution system would require buyers' cooperation either for fingerprint embedding (as in the proposal of Chapter 2) or redistributor tracing (as in the protocols of Chapter 3).

Acknowledgments and disclaimer

Thanks go to Jordi Soria for useful observations about the security and privacy of the protocols described in Chapter 2. This work was partly funded by the European Commission under FP7 projects "DwB" and "Inter-Trust", by the Spanish Government through projects TSI2007-65406-C03-01/03 "E-AEGIS", TIN2011-27076-C03-01/02 "CO-PRIVACY", and CONSOLIDER INGENIO 2010 CSD2007-0004 "ARES", and by the Government of Catalonia through grant 2009 SGR 1135. Dr. Josep Domingo-Ferrer is partly supported as an ICREA-Acadèmia researcher by the Government of Catalonia. He holds the UNESCO Chair in Data Privacy, but the views expressed in this report are his own and do not commit UNESCO.

Bibliography

[1] BitTorrent, http://www.bittorrent.com Accessed on October 23, 2012.

[2] eDonkey2000, http://edonkey2000.co.nr Accessed on October 23, 2012.

[3] Opera software by Opticom. http://www.opticom.de/products/opera.html. Accessed on October 23, 2012.

[4] Pando Networks. http://www.pandonetworks.com/p2p. Accessed on October 23, 2012.

[5] EBU SQAM - Sound Quality Assessment Material. ftp://ftp.tnt.uni-hannover.de/pub/MPEG/audio/sqam/. Accessed on October 23, 2012.

[6] Arora, G., Hannegham, M. and Merabti, M.. P2P commercial digital content exchange. *Electronic Commerce Research and Applications* 4:250-263, 2005.

[7] Blakley, G. R., Meadows, C., and Purdy, G. B. Fingerprinting long forgiving messages. In *Advances in Cryptology-CRYPTO '85*, LNCS 218, pp. 180-189. Berlin-Heidelberg: Springer, 1986.

[8] Bo, Y., Piyuan, L., and Wenzheng, Z. An efficient anonymous fingerprinting protocol. In *Computational Intelligence and Security*, LNCS 4456, pp. 824-832. Berlin-Heidelberg: Springer, 2007.

[9] Boneh, D. and Shaw, J. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897-1905, 1998.

[10] Camenisch, J. Efficient anonymous fingerprinting with group signatures. In *Advances in Cryptology - ASIACRYPT 2000*, LNCS 1976, pp. 415-428. Berlin-Heidelberg: Springer, 2000.

[11] Chang, C.-C., Tsai, H.-C., and Hsieh, Y.-P. An efficient and fair buyer-seller fingerprinting scheme for large scale networks. *Computers & Security*, 29(2):269-277, 2010.

[12] Chaum, D. L.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM **24**(2):84–90 (1981)

[13] Chaum, D.: Untraceable electronic cash. In *Advances in Cryptology- CRYPTO '88*, LNCS 403, pp. 319-327 (1990).

[14] Chaum, D., Damgård, I., van de Graaf, J.: Multiparty computations ensuring privacy of each party's input and correctness of the result. In Advances in Cryptology-CRYPTO'87, LNCS 293, Springer, pp. 87–119 (1988)

[15] Cox, I. J., Miller, M. L., Bloom, J. A. , Fridrich, J., Kalker, T.: Digital Watermarking and Steganography. Burlington MA: Morgan Kaufmann (2008)

[16] Damgård, I., Ishai, Y., Krøigaard, M.: Perfectly secure multiparty computation and the computational overhead of cryptography. In EUROCRYPT 2010, LNCS 6110, Springer, pp. 445–465 (2010)

[17] Domingo-Ferrer, J.: Anonymous fingerprinting of electronic information with automatic identification of redistributors. Electronics Letters, **34**(13):1303–1304 (1998)

[18] Domingo-Ferrer, J.: Anonymous fingerprinting based on committed oblivious transfer. In Public Key Cryptography-PKC 1999, LNCS 1560, Springer, pp. 43–52 (1999)

[19] Domingo-Ferrer, J.: Coprivacy: towards a theory of sustainable privacy. In Privacy in Statistical Databases-PSD 2010, LNCS 6344, Springer, pp. 258–268 (2010)

[20] Domingo-Ferrer, J.: Coprivacy: an introduction to the theory and applications of co-operative privacy. SORT-Statistics and Operations Research Transactions, **35**(special issue: Privacy in statistical databases):25–40 (2011)

[21] Domingo-Ferrer, J., Herrera-Joancomartí, J.: Short collusion-secure fingerprints based on dual binary Hamming codes. Electronics Letters **36**(20):1697–1699 (2000)

[22] Domingo-Ferrer, J., Megías, D. Distributed Multicast of Fingerprinted Content Based On a Rational Peer- to-Peer Community. (Submitted).

[23] Fallahpour, M. and Megías, D.: High capacity audio watermarking using the high frequency band of the wavelet domain. Multimedia Tools and Applications **52**(2):485–498 (2011)

[24] Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Boston: Addison-Wesley (1989)

[25] ITU-R. Recommendation BS.1387. Method for objective measurements of perceived audio quality, Dec. 1987.

[26] Katzenbeisser, S., Lemma, A., Celik, M., van der Veen, M., and Maas, M. A buyer-seller watermarking protocol based on secure embedding. *IEEE Transactions on Information Forensics and Security*, 3(4):783-786, 2008.

[27] Kuribayashi, M. On the implementation of spread spectrum fingerprinting in asymmetric cryptographic protocol. *EURASIP Journal on Information Security*, 2010:1:1-1:11, Jan. 2010.

[28] Kuribayashi, M. and Tanaka, H.. Fingerprinting protocol for images based on additive homomorphic property. *IEEE Transactions on Image Processing*, 14(12):2129-2139, Dec. 2005.

[29] Lei, C.-L-, Yu, P.-L., Tsai, P.-L., Chan, M.-H.: An efficient and anonymous buyer-seller watermarking protocol. IEEE Transactions on Image Processing, **13**(12):1618–1626 (2004)

[30] Mas-Colell, A., Whinston, M., and Green, J. *Microeconomic theory*. New York NY: Oxford University Press, 1995.

[31] Maymounkov, P., Mazières. D.: Kademlia: a peer-to-peer information system based on the XOR metric. In PTPS 2002-First International Workshop on Peer-to-Peer Systems, LNCS 2429, Springer, pp. 43–65 (2002)

[32] Megías, D., Domingo-Ferrer, J.: Privacy-Aware Peer-to-Peer Content Distribution Using Automatically Re-combined Fingerprints. (Submitted).

[33] Megías, D., Serra-Ruiz, J., Fallahpour, M.: Efficient self-synchronised blind audio watermarking system based on time domain and FFT amplitude modification. Signal Processing, **90**(12):3078–3092 (2010)

[34] Memon, N., Wong, P. W.: A buyer-seller watermarking protocol. IEEE Transactions on Image Processing, **10**(4):643–649 (2001)

[35] Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V., editors. *Algorithmic Game Theory*. New York NY: Cambridge University Press, 2007.

[36] Nuida, K., Fujitsu, S., Hagiwara, M., Kitagawa, T., Watanabe, H., Ogawa, K., Imai, H.: An improvement of Tardos's collusion-secure fingerprinting codes with very short lengths. In Proceedings of the 17th international conference on Applied algebra, algebraic algorithms and error-correcting codes (AAECC'07). Springer, pp. 80–89 (2007)

[37] Pfitzmann, B., Waidner, M.: Anonymous fingerprinting. In Advances in Cryptology-EUROCRYPT'96, LNCS 1233, Springer, pp. 88–102 (1997)

[38] Pfitzmann, B., Sadeghi, A.-R.: Coin-based anonymous fingerprinting. In Advances in Cryptology-EUROCRYPT'99, LNCS 1592, Springer, pp. 150–164 (1999)

[39] Pfitzmann, B., Sadeghi, A.-R.: Anonymous fingerprinting with direct non-repudiation. In *Advances in Cryptology- ASIACRYPT 2000*, pp. 401-414. Berlin-Heidelberg: Springer-Verlag, 2000.

[40] Prins, J. P., Erkin, Z., Lagendijk, R. L.: Anonymous fingerprinting with robust QIM watermarking techniques. EURASIP Journal on Information Security, **2007**:20:1–20:7 (2007)

[41] Rivest, R. L., and Shamir, A. PayWord and MicroMint: Two simple micropayment schemes. Technical report, MIT LCS, November 1995.

[42] Tardos, G.: Optimal probabilistic fingerprint codes. In Proceedings of the thirty-fifth annual ACM symposium on Theory of computing (STOC '03). ACM, pp. 116–125 (2003)

[43] Thiede, T., Treurniet, W. C., Bitto, R., Schmidmer, C., Sporer, T., Beerends, J. G., and Colomes, C. PEAQ - the ITU standard for objective measurement of perceived audio quality. *Journal of the Audio Engineering Society*, 48(1/2):3-29, 2000.