Systematic Generation of Tardos's Fingerprint Codes

Kuribayashi, Minoru

Morii, Masakatu

(Citation)
IEICE transactions on fundamentals of electronics, communications and computer sciences,E93-A(2):508-515

(Issue Date)
2010-02-01

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90001352

PAPER

Systematic Generation of Tardos's Fingerprint Codes

Minoru KURIBAYASHI[†a)] *and* **Masakatu MORII**[†], *Members*

SUMMARY Digital fingerprinting is used to trace back illegal users, where unique ID known as digital fingerprints is embedded into a content before distribution. On the generation of such fingerprints, one of the important properties is collusion-resistance. Binary codes for fingerprinting with a code length of theoretically minimum order were proposed by Tardos, and the related works mainly focused on the reduction of the code length were presented. In this paper, we present a concrete and systematic construction of the Tardos's fingerprinting code using a chaotic map. Using a statistical model for correlation scores, the actual number of true-positive and false-positive detection is measured. The collusion-resistance of the generated fingerprinting codes is evaluated by a computer simulation.

key words: *fingerprinting, Tardos's code, chaotic map, collusion attack*

1. Introduction

Digital fingerprinting is a method to insert user's own ID into digital contents in order to identify illegal users who distribute unauthorized copies. One of the serious problems in a fingerprinting system is the collusion attack such that several users combine their copies of a same content to modify/delete the embedded fingerprints. In order to design a collusion-resistant fingerprint, two kinds of approaches have been studied. One approach is to exploit the spread spectrum (SS) technique [1]–[4], and the other approach is to devise an exclusive code, known as collusion-resistant code [5]–[10], which can trace colluders.

An early work on designing collusion-resistant binary fingerprinting codes was presented by Boneh and Shaw [5] underlying the principle referred to as the *marking assumption*. In this case, a fingerprint is a set of redundant digits which are distributed in some random positions of an original content. When a coalition of users attempts to discover some of the fingerprint positions by comparing their marked copies for differences, the coalition may modify only those positions where they find a difference in their fingerprinted copies. A c-secure code guarantees the tolerance for the collusion attack with c pirates or less. Tardos [10] has proposed a probabilistic c-secure code with error probability ϵ_1 which has a length of theoretically minimal order with respect to the number of colluders. On the binary digits of the codeword, the frequency of "0" and "1" is ruled by a specific probability distribution referred to as the *bias distribution*.

The code length of Tardos's code has been reduced by a strict evaluation of tracing error probabilities [11] and modifying some parameters using statistical properties [12]. However, there are still problems such as the large required memory and impossibility of explicit implementation, which are mainly due to the continuity of probability distributions used in the codeword generation. Instead of continuous distributed probability, finite probability distributions were explored in [13] in order to reduce the required memory as well as the code length.

In this paper, we present an explicit construction method of Tardos's codes using a well-known chaotic map, *logistic map* [14], showing the relation among the probability distributions of Tardos's codes and the distribution of chaotic sequences. We also study the statistical property of correlation scores, and present the actual number of true-positive and false-positive using a proper threshold for determining colluders. In order to generate the chaotic sequence on a computer, the precision becomes a bottleneck because it may output a certain periodic values and may converge on one value. If the number of digits for representing the sequence is sufficiently large, such cases can be neglected. Different from the related works of Tardos's code, our interest is mainly in the ease of implementation on a computer. The chaotic map provides us a simple solution for calculating the probability distribution from a given initial value, which can be secret information in the fingerprinting system. Strictly speaking, our constructed code is modified version of Tardos's one.

2. Preliminaries

In this section, we first review the original Tardos's binary fingerprinting code [10], and then show the related works. Furthermore, we review a well-known chaotic map which is analogously related to the parameters of the code.

2.1 Tardos's Code

Let N be the allowable number of users in a fingerprinting system. The Tardos's fingerprinting scheme distributes a binary codeword of length L to each user. The codewords are arranged as an $N \times L$ matrix $\boldsymbol{X}$, where the j-th row corresponds to the fingerprint given to the j-th user. The generation of the matrix $\boldsymbol{X}$ is composed of two steps.

1. A distributor is supposed to choose the random variables $0 < p_i < 1$ independently for every $1 \leq i \leq L$,

Manuscript received May 26, 2009.
Manuscript revised August 20, 2009.

[†]The authors are with the Graduate School of Engineering, Kobe University, Kobe-shi, 657-8501 Japan.
a) E-mail: kminoru@kobe-u.ac.jp

DOI: 10.1587/transfun.E93.A.508

according to a given bias distribution $\mathcal{P}$, which satisfies the following conditions.

- $t = 1/300c$
- $0 < t' < \pi/4, \quad \sin^2 t' = t, \quad r_i \in [t', \pi/2 - t']$
- $p_i = \sin^2 r_i, \quad t \le p_i \le 1 - t$

Here r_i is uniformly and randomly selected from the above range.

2. Each entry $X_{j,i}$ of the matrix $\boldsymbol{X}$ is selected independently from the binary alphabet $\{0, 1\}$ with $\Pr(X_{j,i} = 1) = p_i$ and $\Pr(X_{j,i} = 0) = 1 - p_i$ for every $1 \le j \le N$.

Let C be a set of colluders and c be the number of colluders. Then we denote by $\boldsymbol{X_C}$ the $c \times L$ matrix of codewords assigned to the colluders. Depending on the attack strategy ρ, which is called C-strategy, the fingerprint $\boldsymbol{y} = (y_1, \ldots, y_L), y_i \in \{0, 1\}$ contained in a pirated copy is denoted by $\boldsymbol{y} = \rho(\boldsymbol{X_C})$. In a tracing (accusation) algorithm $\mathcal{A}$, a correlation score S_j of the j-th user is calculated

$$S_j = \sum_{i=1}^{L} y_i U_{j,i}, \tag{1}$$

where

$$U_{j,i} = \begin{cases} \sqrt{\frac{1-p_i}{p_i}} & \text{if } X_{j,i} = 1 \\ -\sqrt{\frac{p_i}{1-p_i}} & \text{if } X_{j,i} = 0. \end{cases} \tag{2}$$

If S_j exceeds a threshold Z, the j-th user is decided as guilty. The algorithm $\mathcal{A}$ outputs a list of suspicious users. For convenience, we denote the correlation score of an innocent user "$\bar{j}$" by $S_{\bar{j}}$ and that of a fixed colluder by S.

The Tardos's fingerprinting scheme uses a code length L and a threshold Z with the following scaling behavior as a function of N, c, and a false-positive parameter ϵ_1:

$$L = 100c^2 \lceil \ln 1/\epsilon_1 \rceil, \tag{3}$$

$$Z = 20c \lceil \ln 1/\epsilon_1 \rceil, \tag{4}$$

where $\lceil x \rceil$ denotes the smallest integer that is not less than x.

At the collusion attack, c colluders try to detect the positions of the embedded codeword from differences of their copies, and then to modify bits of the codeword in these positions. This attack model is formulated as the following assumption;

Assumption 1: (Marking Assumption): Let we say that position i is undetectable for C if the codewords assigned to c colluders in C match in i-th position. Then, $y_i = X_{j,i}$ for any $j \in C$.

Under the marking assumption, C-strategy satisfies the following assumption.

Assumption 2: Colluders have no information on the i-th position of innocent users if it is undetectable.

On the evaluation of a fingerprinting code, the false-positive probability, denoted by $\Pr[FP]$, and false-negative probability, denoted by $\Pr[FN]$, are bounded as follows.

Definition 1: (Soundness): Let $\epsilon_1 \in (0, 1)$ be a fixed constant and let j be an arbitrary innocent user. We say that a fingerprinting scheme is ϵ_1-sound if, for any C-strategy ρ, the probability $\Pr[FPj]$ that an innocent user j gets accused is bounded by

$$\Pr[FPj] = \Pr[j \in \mathcal{A}(\rho(\boldsymbol{X_C}))] < \epsilon_1. \tag{5}$$

Definition 2: (Completeness): Let $\epsilon_2 \in (0, 1)$ be a fixed constant. We say that a fingerprinting scheme is ϵ_2-complete if, for any C-strategy ρ, the false-negative probability $\Pr[FN]$ that no colluder gets accused is bounded by

$$\Pr[FN] = \Pr[C \cap \mathcal{A}(\rho(\boldsymbol{X_C})) = \emptyset] < \epsilon_2. \tag{6}$$

The original Tardos's code is ϵ_1-sound for any coalition C of arbitrary size and ϵ_2-complete for any coalition C of size at most c and $\epsilon_2 = \epsilon_1^{c/4}$. If N is larger than 2^{L+1}, most users must share their codewords with another user, thus even if a single user distributes his copy, another user will be accused. The above bounds are not strict, there is a room for improvement.

The false-positive probability $\Pr[FP]$ that some innocent users are accused is given by $\epsilon = \epsilon_1 N$. It is shown that Tardos's fingerprinting code has an error probability ϵ at most as follows;

Proposition 1: The association of false-positive and false-negative probabilities are given by the inequality

$$\Pr[FP \text{ or } FN] \ \le \ \Pr[FP] + \Pr[FN] < \epsilon, \tag{7}$$

when $c \ge 4$.

Here we give the proof for completeness. There are $(N - c)$ innocent users, and a bound of false-positive probability $\Pr[FP]$ that some innocent users are accused is given by

$$\begin{aligned} \Pr[FP] &= 1 - (1 - \Pr[FPj])^{N-c} \\ &\le (N - c) \Pr[FPj] < (N - 1)\epsilon_1. \end{aligned} \tag{8}$$

By Eq. (6) and Eq. (8),

$$\begin{aligned} \Pr[FP] + \Pr[FN] &\le (N - c) \Pr[FPj] + \Pr[FN] \\ &< (N - 1)\epsilon_1 + \epsilon_1^{c/4} < \epsilon. \end{aligned} \tag{9}$$

Since $\epsilon_1^{c/4} \le \epsilon_1$ is true only when $c \ge 4$, so Eq. (7) is derived.

Note that the evaluation in Proposition 1 is not always sharp. Namely, the inequality in Eq. (8) is motivated by the approximation of binomial theorem

$$\begin{aligned} (1 - x)^n &= 1 - {}_nC_1 x + {}_nC_2 x^2 - {}_nC_3 x^3 \ldots \\ &\ge 1 - nx, \end{aligned} \tag{10}$$

where $0 \le x \ll 1$, however, the approximation is not sharp when n is too large relative to x^{-1}. Therefore, when the product $(N - c) \Pr[FPj]$ becomes large, the bound given by Proposition 1 becomes far from being sharp, e.g., the bound of probability can even exceed 1 in some cases. In this paper, an actual number of true-positive and false-positive detection is measured for the evaluation of fingerprinting code.

2.2 Related Works

The main concern in the related works is to reduce the code length L. If one assumes that the scores S_j and S are modeled by Gaussian distribution, then there is the shortest length code achieving the $\Pr[FPj]$ and $\Pr[FN]$. Škorić et al. [12] assumed that the correlation scores S_j and S have the probability density modeled by Gaussian distribution using the Central Limit Theorem: when a large number of i.i.d. variables are summed, the distribution of the sum converges to the normal distribution. The tracing algorithm $\mathcal{A}$ computes the sum of $y_i U_{j,i}$ over L independent terms, and all the terms have the same bias distribution. Under the Gaussian approximation, the statistical analysis of the Tardos's code provides the lower bound on the code length.

Some hurdles in the fingerprinting system are the large required memory and impossibility of explicit implementation, which are mainly due to the continuity of bias distributions used in the codeword generation. Nuida et al. [13] focused on the bias distributions used in the codeword generation, and provided the discrete bias distributions for the purpose of reducing the required memory amount. They showed that the optimal distribution has only $\lceil c/2 \rceil$ possible outputs; thus only $\lceil \log_2 \lceil c/2 \rceil \rceil$-bits memory is required to record one output. However, the required memory to record all bias distribution is still increased by a factor of L; thus it is $\lceil \log_2 \lceil c/2 \rceil \rceil \times L$.

As given in Eq. (1), the scores S_j and S are defined as the sum of a large number of stochastic variables. So it is expected that S_j and S are to have an approximately Gaussian probability distribution. The Gaussian approximation of S_j and S has been presented in [12], and a strict evaluation of code length and threshold was investigated under a designed false-positive probability ϵ_1. Instead of the probability, we calculate the actual number of true-positive and false-positive detection using a properly designed threshold based on the Central Limit Theorem and the statistical property.

Let $a_i, (1 \leq i \leq n)$ be a set of n i.i.d. random variables having finite values of mean μ and variance $\sigma^2 > 0$. The Central Limit Theorem states that as the sample size n increases, the distribution of the sum approaches the normal distribution $N(n\mu, n\sigma^2)$, where $n\mu$ is a mean and $n\sigma^2$ is a variance.

The correlation score S_j is the result of sum of $U_{j,i}$ only when $y_i = 1$. Škorić et al. [12] explored the evaluation of the score from the viewpoint of the symmetric property of $U_{j,i}$, and modified the correlation score $\hat{S}_j$ as follows.

$$\hat{S}_j = \sum_{i=1}^{L} (2y_i - 1) U_{j,i} \tag{11}$$

Due to this modification, the distribution of $\hat{S}_j$ approaches $N(0, L)$ because all L elements are contributed on the score. The modification is also done for the correlation score of colluders S, and we denote it by $\hat{S}$. It is noted that the Gaussian approximation is also valid for the modified scores $\hat{S}_j$ and $\hat{S}$.

Furon et al. studied the statistics of the score $\hat{S}_j$ and $\hat{S}$ [15]. In the study, without loss of generality, the mean and variance of $\hat{S}_j$ are 0 and L, and those of $\hat{S}$ are $2L/c\pi$ and $L(1 - 4/c^2\pi^2)$, respectively. In the paper, they insisted that the use of the Central Limit Theorem was absolutely not recommended when estimating the code length because it amounts to integrate the distribution function on its tail where the Gaussianity assumption does not hold. The Berry-Esséen bound shows that the gap between the Gaussian law and the real distribution of the scores depends on their third moment, which a priori depends on the collusion process [11]. However, it is reported in [11] that the approximation of the distribution by a Gaussian is accurate starting from a value of c between 10 and 20, and the derived code length becomes a factor 2 shorter than the estimation using Eq. (11). Considering a practical use, it is advisable to apply the approximation.

2.3 Chaotic Map

We found that the bias distribution $\mathcal{P}$ of Tardos's code is analogously represented by a well-known chaotic map: *logistic map* [14].

The logistic map is the simplest known non-linear dynamical system, and possibly the most studied such system, which has perfect chaotic properties. The logistic map is a one-dimensional discrete-time system described by the nonlinear difference equation,

$$P_{i+1} = aP_i(1 - P_i), \qquad (0 < P_i < 1), \tag{12}$$

where a is constrained by $0 < a \leq 4$, and the behavior of P_i is sensitive to the value of a. For $a < a^\star = 3.5699456\ldots$, the orbit of P_i is periodic between the several values corresponding to a. For $a > a^\star$ chaotic behavior sets in, interspersed by finite intervals of stable periodic behavior, and when $a = 4$, it is completely chaotic. In the rest of this paper, we set $a = 4$ for the generation of the chaotic sequences.

Generally, a decimal number is treated as floating point number on a computer and sometimes the precision is sacrificed for reducing the computing costs. For the countermeasure of such a problem, the range of the chaotic sequence generated by logistic map is transformed from decimal number to integer. The modified map is calculated as follows.

$$\tilde{P}_{i+1} = 4\tilde{P}_i(2^T - \tilde{P}_i)/2^T, \ (0 < \tilde{P}_i < 2^T) \tag{13}$$

And Eq. (13) is simplified further by bit-shift operation.

$$\tilde{P}_{i+1} = \tilde{P}_i(2^T - \tilde{P}_i) \gg T - 2 \tag{14}$$

Here "$\gg \alpha$" denotes α-bit-shift operation to the right. The value of original logistic map, P_i, is approximately recovered from $\tilde{P}_i$ by dividing the value by 2^T. So, by an inverse transform from $\tilde{P}_i$ to P_i, namely $P_i = \tilde{P}_i/2^T$, the desired chaotic sequence with T-bit precision can be obtained.

3. Proposed Systematic Implementation

Under the assumption that the probability density function of scores $\hat{S}_j$ and $\hat{S}$ follows Gaussian distribution both for innocent users and colluders, we discuss the stochastic approach for calculating the actual number of true-positive and false-positive detection using a threshold $\hat{Z}$ and the analogy of the bias distribution.

3.1 Design of Threshold

From the viewpoint of a false-positive error, the statistical property of a normal distribution helps us to design a proper threshold $\hat{Z}(\leq Z)$. The false-positive probability $\Pr[FPj]$, which is the probability such that $\hat{S}_j$ exceeds $\hat{Z}$, is given by

$$\Pr[FPj] = \Pr(\hat{S}_j > \hat{Z}) = \frac{1}{2}\mathrm{erfc}\left(\frac{\hat{Z}}{\sqrt{2L}}\right), \tag{15}$$

where "erfc" stands for the complementary error function. Using the inverse function of erfc, we can get the threshold $\hat{Z}$ as follows.

$$\hat{Z} = \sqrt{2L}\cdot\mathrm{erfc}^{-1}(2\Pr[FPj]) = \sqrt{2L}\cdot\mathrm{erfc}^{-1}(2\epsilon_1) \tag{16}$$

The design of threshold $\hat{Z}$ based on the statistical approach is studied in [12]. By imposing, for example, that $\epsilon_1 = 10^{-8}$, we find

$$\hat{Z} = 3.97\sqrt{2L}. \tag{17}$$

On the other hand, using Eq. (3) and Eq. (4), the original threshold Z is modified by

$$Z = 20\sqrt{\frac{L}{100\lceil\ln 1/\epsilon_1\rceil}}\times\lceil\ln 1/\epsilon_1\rceil = 2\sqrt{L\lceil\ln 1/\epsilon_1\rceil}, \tag{18}$$

and under the parameter $\epsilon_1 = 10^{-8}$ we obtain

$$Z = 8.72\sqrt{L}. \tag{19}$$

If a threshold $\hat{Z}$ is calculated with respect to a given probability ϵ_1, then the probability of true-positive per each colluder P_{TP} can be calculated. It is the area of probability density function of $\hat{S}$ which value exceeds $\hat{Z}$. From Sect. 2.2, it follows $N(2L/c\pi, L(1-4/c^2\pi^2))$, hence,

$$P_{TP} = \frac{1}{2}\mathrm{erfc}\left(\frac{1}{\sqrt{2\sigma_c^2}}\left(\hat{Z} - \frac{2L}{c\pi}\right)\right), \tag{20}$$

where

$$\sigma_c^2 = L\left(1 - \frac{4}{c^2\pi^2}\right). \tag{21}$$

Therefore, the expected number of detectable colluders N_{TP} is

$$N_{TP} = cP_{TP} = \frac{c}{2}\mathrm{erfc}\left(\frac{1}{\sqrt{2\sigma_c^2}}\left(\hat{Z} - \frac{2L}{c\pi}\right)\right), \tag{22}$$

and the false-negative probability $\Pr[FN](=(P_{FN})^c)$, where P_{FN} is the probability of false-negative per each colluder, is

$$\begin{aligned}\Pr[FN] &= (P_{FN})^c \\ &= \left(1 - P_{TP}\right)^c \\ &= \left(1 - \frac{1}{2}\mathrm{erfc}\left(\frac{1}{\sqrt{2\sigma_c^2}}\left(\hat{Z} - \frac{2L}{c\pi}\right)\right)\right)^c.\end{aligned} \tag{23}$$

The actual number of false positive N_{FP}, which is the average number of detected innocent users at each detection, is

$$N_{FP} = (N-c)P_{FP} = (N-c)\epsilon_1. \tag{24}$$

A sketch of the probabilities P_{TP}, P_{FP}, and P_{FN} is illustrated in Fig. 1.

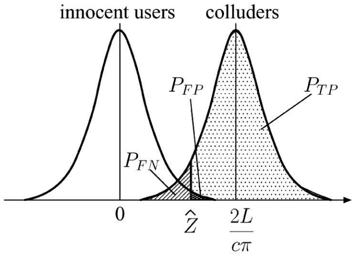


Fig. 1 The probability density function of correlation scores $\hat{S}_j$ and $\hat{S}$.

3.2 Generation of Bias Distribution

Let $P_0 = \sin^2 R_0$, $(0 < R_0 < \pi)$ be an initial value of a chaotic sequence. Then the i-th element is given by

$$P_i = \sin^2 R_i,\ (0 < R_i < \pi), \tag{25}$$

and the variables R_i is represented by

$$R_{i+1} = \begin{cases} 2R_i & 0 < R_i \leq \pi/2 \\ 2\pi - 2R_i & \pi/2 < R_i < \pi \end{cases} \tag{26}$$

It is noted that the chaotic behavior of P_i is broken when $R_i = \pi/2$. However, if the values P_i have infinite precision and the initial value R_0 is chosen uniformly at random, then the probability that R_i takes $\pi/2$ is 0. Notice that the sequence R_i also follows a well-known chaotic map: *tent map* [14]. Here, we suppose T' which satisfies

$$\sin^2(1/2^{T'}) = 1/2^T. \tag{27}$$

Then the sequence R_i is uniformly distributed in the range $[1/2^{T'}, \pi - 1/2^{T'}]$. Since the orbit of the tent map is symmetric with respect to the axis $x = \pi/2$, the range of R_i can be simplified to $[1/2^{T'}, \pi/2 - 1/2^{T'}]$. The parameter $1/2^{T'}$ is corresponding to t' if it satisfies $0 < 1/2^{T'} < \pi/4$. By setting the precision T carefully, the remaining condition

$$1/2^T = 1/300c, \tag{28}$$

can be fulfilled. Therefore, the chaotic sequence P_i satisfies the conditions of p_i shown at the generation of the matrix $\boldsymbol{X}$,

Table 1 The comparison of required memory for the bias distribution.

Original [10]	Nuida [13]	Proposed
$32L$	$\lceil\log_2\lceil c/2\rceil\rceil L$	T

where each parameter denoted by a capital letter is equivalent to corresponding the parameter of the Tardos's code. It is noted that Eq. (28) cannot be perfectly fulfilled by an integer T since $300c$ is not a power of 2. In this regard, our constructed code is different from the original Tardos's one.

The bias distribution p_i can be generated by the logistic map with an initial value P_0. It is worth mentioning that the required memory is constant and much smaller than the original and modified Tardos's codes [10], [13].

Let $\tilde{P}_0$ be an initial value for the transformed logistic map mentioned in Sect. 2.3. We denote by $f^i(\tilde{P}_0)$ the generated random variables for every $1 \leq i \leq L$ using the logistic map, which have one-to-one correspondence with p_i of the Tardos's code. For the generation of a codeword for each user, a distributor uniformly and independently chooses random variables $x_i \in [0, 2^T]$. There is a random function with a secret key *key* which outputs x_i; an example is a pseudorandom number generator (PRNG). Then each entry $X_{j,i}$ of the matrix $\boldsymbol{X}$ is determined as

$$X_{j,i} = \begin{cases} 1 & x_i \leq f^i(\tilde{P}_0), \\ 0 & x_i > f^i(\tilde{P}_0). \end{cases} \tag{29}$$

The secrecy of $f^i(\tilde{P}_0)$ and x_i for every $1 \leq i \leq L$ is solely dependent on that of P_0 and *key* under the assumption that the sensitivity of initial state in the chaotic map is intractable and the applied PRNG is secure. We can employ the logistic map for PRNG, though it is out of our research scope.

Table 1 shows the required memory to record the bias distribution, where the precision for representing the probability p_i is approximated by single-precision binary floating point numbers. Since only the initial value P_0 of T-bits is stored in our method, the required memory is constant.

4. Numerical Results

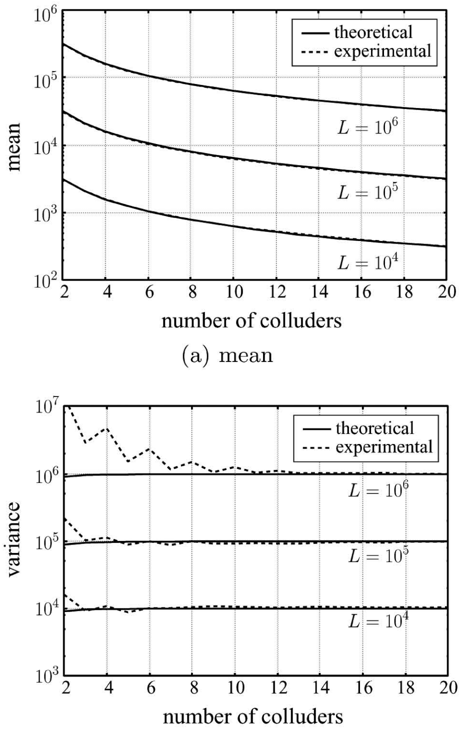
We implement the proposed construction method and evaluate the collusion-resistance. Without loss of generality, c codewords are randomly chosen to produce a pirated codeword $\boldsymbol{y}$. Under the marking assumption, if i-th bit of c codewords is different, that of pirated codeword y_i is selected by the following 5 kinds of collusion manner;

- majority: If the sum of i-th bit exceeds $c/2$, $y_i = 1$, otherwise, $y_i = 0$.
- minority: If the sum of the i-th bit exceeds $c/2$, $y_i = 0$, otherwise, $y_i = 1$.
- random: $y_i \in_R \{0, 1\}$.
- all 0: $y_i = 0$.
- all 1: $y_i = 1$.

We attempt to detect the codewords of colluders as many as possible using the threshold $\hat{Z}$. It is remarkable that a well-known collusion attack is averaging c copies and is equivalent to "majority" attack if a detector rounds decimal numbers to the nearest symbols "0" and "1." Hence, the following experiments are derived under the majority attack.

Table 2 The estimation of the distribution of $\hat{S}_j$ with the number of users $N = 10^4$.

length L	mean	variance
1×10^4	0.28	9,977.36
1×10^5	−0.63	100,043.59
1×10^6	−5.12	997,897.87



(a) mean

(b) variance

Fig. 2 The evaluation of the distribution of $\hat{S}$.

4.1 Distribution of $\hat{S}_j$ and $\hat{S}$

For the evaluation of the validity of our design of a threshold $\hat{Z}$, we calculate the mean and variance of correlation scores $\hat{S}_j$ except for the colluders' ones. The precision parameter of logistic map is set $T = 30$, and the number of users is fixed to $N = 10^4$ in this simulation. The collusion model is majority attack. Table 2 shows the results of various code lengths, where the results are the average values of 10^2 trials. We can see that the distribution of $\hat{S}_j$ is approximately $N(0, L)$. We also show the mean and variance of colluders' scores in Fig. 2, where the solid lines are theoretical figures calculated by $N(2L/c\pi, L(1 - 4/c^2\pi^2))$ and dot ones are experimental results. Although the variance oscillates when the number of colluders is small, the correlation score of implemented fingerprinting codes almost follow the theoretical distribution.

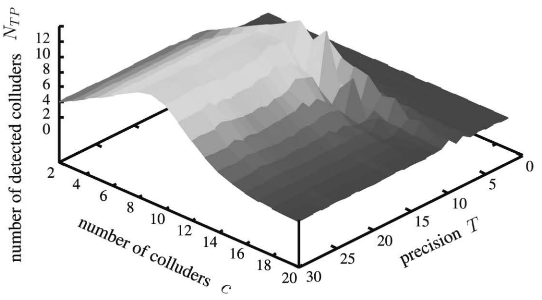


Fig. 3 The average number of detected colluders N_{TP} with each precision T.

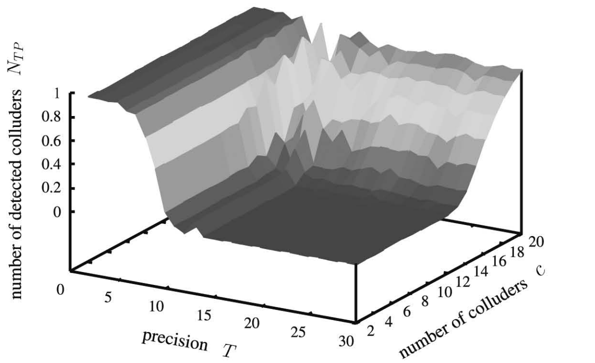


Fig. 4 The false-negative probability Pr[FN] with each precision T.

4.2 Precision T

From the conditions of Tardos's fingerprinting codes, the precision parameter T must satisfy Eq. (28), hence $2^T = 300c$. For example, if $c = 10$, then $T \approx 11.56$. Here, the dynamical degradation of digital chaotic systems realized in finite precision should be considered. The related problems include short cycle-length, non-ideal distribution and correlation, etc. There are only 2^T values to represent the chaotic orbits. So the cycle length of any chaotic orbit will be not larger than 2^T. Some quantization errors will be introduced when the chaotic map is iterated, which makes the dynamics of digital chaotic systems badly depart from the theoretical one. One of the remedies to overcome the degradation is to use higher precision. For the evaluation of the performance with each precision T, N_{TP} and Pr[FN] are examined under the conditions; the code length $L = 10^4$, the number of users $N = 10^4$, the false-positive probability $\epsilon_1 = 10^{-8}$, and the number of trials is 10^3 with differently selected initial values for the chaotic map. The collusion model is majority attack. The results are shown in Fig. 3 and Fig. 4. From the results, when $T \geq 16$, the degradation of the performance is slight. However, from the viewpoint of security, we recommend to use $T \geq 30$ because of the key space for the bias distribution. It is noted that we can not exclude the possibility of short cycle-length even if the precision is increased. By carefully selecting the initial value, such a case can be avoided. Hereafter, the simulation is performed using a fixed precision $T = 30$.

4.3 Evaluation of Our Codes

We implement the proposed generation algorithm and evaluate the number of positively detected colluders, the actual number of false-positive detection N_{FP}, and the false-negative probability Pr[FN]. In the simulation, we fixed the code length $L = 10^4$ and the number of users $N = 10^4$. Figure 5 shows the average number of detected colluders N_{TP} with 10^4 patterns of trials using majority attack. The comparison with Tardos's code is shown in Fig. 6. We also compare the false-negative probability Pr[FN], which results are plotted in Fig. 7 and Fig. 8. We can see that the performance of our code is very close to the theoretical analysis and the original Tardos's code. The results of 5 kinds of collusion attacks are shown in Fig. 9 and Fig. 10. The similar results are derived for the original Tardos's code, hence they are omitted. As the consequence, we can say that there is no distinct difference for those collusion attacks and that our code has the same performance as Tardos's code under these typical 5 attacks.

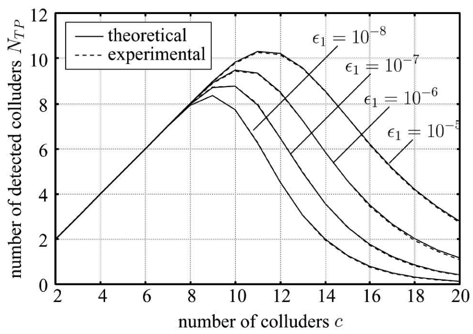


Fig. 5 The average number of detected colluders N_{TP} with each false-positive probability ϵ_1.

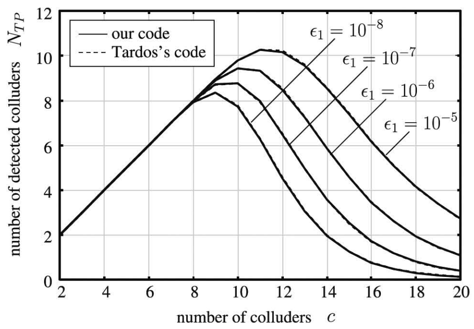


Fig. 6 The comparison of the experimental values of N_{TP}.

Due to the limitation of computational resources, the number of trials for evaluating the false-positive detection is only 10^4 times. Considering the precision of the data, we show the number of false-positive N_{FP} in the cases $\epsilon_1 = 10^{-5}$ and 10^{-6} in Fig. 11. Since the actual number of false-positive N_{FP} is almost equal for different number of colluders, the average N_{FP} for different ϵ_1 is shown in Table 3. It is noted that the theoretical value is $(N - c)\epsilon_1 \approx \epsilon_1 \times 10^4$ under

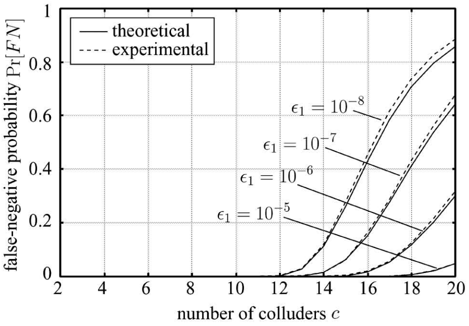


Fig. 7 The false-negative probability Pr[FN] with each false-positive probability ϵ_1.

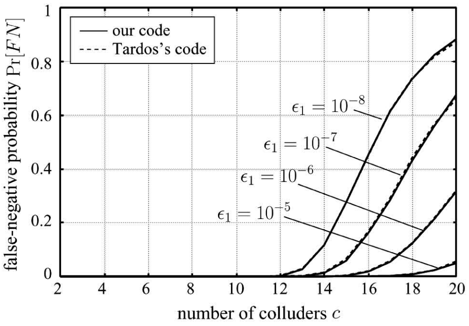


Fig. 8 The comparison of the experimental values of Pr[FN].

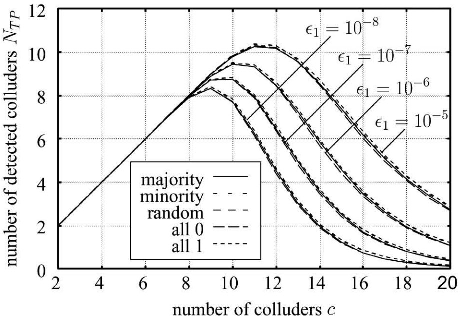


Fig. 9 The comparison of average number of detected colluders N_{TP} for 5 kinds of collusion attacks.

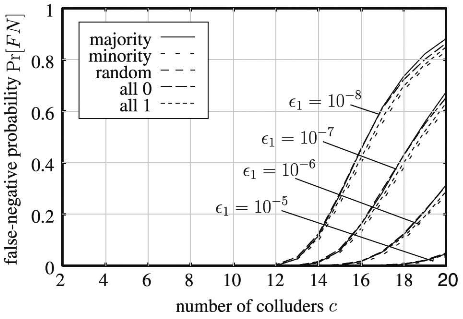


Fig. 10 The comparison of false-negative probability Pr[FN] for 5 kinds of collusion attacks.

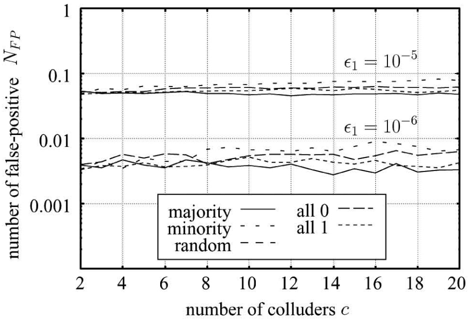


Fig. 11 The comparison of the average number of false-positive N_{FN} for 5 kinds of collusion attacks.

Table 3 The comparison of average number of false-positive N_{FP} for 5 kinds of collusion attacks, where the numbers in parentheses are that of Tardos's code.

collusion	ϵ_1			
attack	10^{-5}	10^{-6}	10^{-7}	10^{-8}
majority	0.489×10^{-1} (0.480×10^{-1})	0.374×10^{-2} (0.377×10^{-2})	0.232×10^{-3} (0.300×10^{-3})	0.105×10^{-4} (0.263×10^{-4})
minority	0.687×10^{-1} (0.669×10^{-1})	0.623×10^{-2} (0.614×10^{-2})	0.595×10^{-3} (0.595×10^{-3})	0.474×10^{-4} (0.789×10^{-4})
random	0.587×10^{-1} (0.524×10^{-1})	0.536×10^{-2} (0.422×10^{-2})	0.405×10^{-3} (0.363×10^{-3})	0.263×10^{-4} (0.263×10^{-4})
all 0	0.597×10^{-1} (0.524×10^{-1})	0.536×10^{-2} (0.422×10^{-2})	0.405×10^{-3} (0.363×10^{-3})	0.263×10^{-4} (0.263×10^{-4})
all 1	0.546×10^{-1} (0.508×10^{-1})	0.418×10^{-2} (0.393×10^{-2})	0.337×10^{-3} (0.305×10^{-3})	0 (0.211×10^{-4})
average	0.583×10^{-1} (0.541×10^{-1})	0.498×10^{-2} (0.445×10^{-2})	0.395×10^{-3} (0.385×10^{-3})	0.221×10^{-4} (0.358×10^{-4})

the above conditions. Compared with the theoretical values, the derived N_{FP} is smaller; when ϵ_1 is 10^{-8}, it is 22%. Though N_{FP} of the original Tardos's code is slightly better than that of our code, the difference can be negligible.

From the above results, we can say that the proposed implementation method successfully generates Tardos's fingerprinting code. However, the actual number of false-positive N_{FP} seems to be lower than the theoretical one. It may come from the reason mentioned in Sect. 2.2 because it amounts to integrate the distribution function of $\hat{S}_j$ on its tail. Optimistically interpreting the experimental results, there is still room to improve the bounds of Pr[FP] and Pr[FN].

It is possible to apply any existing PRNGs whose properties have been well investigated for the generation of bias distribution. For the fulfillment of the conditions subjected to the bias distribution, however, any modification of existing PRNGs should be required, which may degrade the randomness of its output and the performance of generated code. On the other hand, our construction is based on the analogy between the bias distribution and logistic map, and our simulation results confirm the traceability of the constructed fingerprinting code.

5. Conclusions

In this paper, we have discussed about the stochastic approach for designing a threshold for determining colluders and the analogy of the bias distribution of Tardos's fingerprinting code with a logistic map, and proposed a systematic generation method. Our theoretical analysis under the assumption of Gaussianity gives us the actual number of detectable colluders as well as the actual number of false-positive. Different from the conventional schemes, the random variables $p_i, (1 \leq i \leq L)$ is generated from a secretly selected initial value for the logistic map, which reduces the required memory.

Acknowledgment

This research was partially supported by the Ministry of Education, Culture, Sports Science and Technology, Grant-in-Aid for Young Scientists (B) (21760291), 2009.

References

[1] I.J. Cox, J. Kilian, F.T. Leighton, and T. Shamson, "Secure spread spectrum watermarking for multimedia," IEEE Trans. Image Process., vol.6, no.12, pp.1673–1687, 1997.

[2] H.V. Zhao, M. Wu, Z.J. Wang, and K.J.R. Liu, "Forensic analysis of nonlinear collusion attacks for multimedia fingerprinting," IEEE Trans. Image Process., vol.14, no.5, pp.646–661, 2005.

[3] Z.J. Wang, M. Wu, H.V. Zhao, W. Trappe, and K.J.R. Liu, "Anti-collusion forensics of multimedia fingerprinting using orthogonal modulation," IEEE Trans. Image Process., vol.14, no.6, pp.804–821, 2005.

[4] Z.J. Wang, M. Wu, W. Trappe, and K.J.R. Liu, "Group-oriented fingerprinting for multimedia forensics," EURASIP J. Appl. Signal Process., no.14, pp.2142–2162, 2004.

[5] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," IEEE Trans. Inf. Theory, vol.44, no.5, pp.1897–1905, 1998.

[6] W. Trappe, M. Wu, Z.J. Wang, and K.J.R. Liu, "Anti-collusion fingerprinting for multimedia," IEEE Trans. Signal Process., vol.51, no.4, pp.1069–1087, 2003.

[7] Y. Yacobi, "Improved Boneh-Shaw content fingerprinting," Proc. CT-RSA, LNCS, vol.2020, pp.378–391, Springer-Verlag, 2001.

[8] J.N. Staddon, D.R. Stinson, and R. Wei, "Combinatiorial properties of frameproof and traceability codes," IEEE Trans. Inf. Theory, vol.47, no.3, pp.1042–1049, 2001.

[9] Y. Zhu, D. Feng, and W. Zou, "Collusion secure convolutional spread spectrum fingerprinting," IWDW 2005, LNCS, vol.3710, pp.67–83, Springer, Heidelberg, 2005.

[10] G. Tardos, "Optimal probabilistic fingerprint codes," J. ACM, vol.55, no.2, pp.1–24, 2008.

[11] B. Škorić, S. Katzenbeisser, and M. Celik, "Symmetric Tardos fingerprinting codes for arbitrary alphabet sizes," Des., Codes Cryptogr., vol.46, no.2, pp.137–166, 2008.

[12] B. Škorić, T.U. Vladimirova, M. Celik, and J.C. Talstra, "Tardos fingerprinting is better than we thought," IEEE Trans. Inf. Theory, vol.54, no.8, pp.3663–3676, 2008.

[13] K. Nuida, M. Hagiwara, H. Watanabe, and H. Imai, "Optimization of Tardos's fingerprinting codes in a viewpoint of memory amount," IH 2007, LNCS, vol.4567, pp.279–293, Springer, Heidelberg, 2008.

[14] S.C. Phatak and S.S. Rao, "Logistic map: A possible random-number generator," Physical Rev. E, vol.51, no.4, pp.3670–3678, 1995.

[15] T. Furon, A. Guyader, and F. Cérou, "On the design and optimization of Tardos probabilistic fingerprinting codes," IH 2008, LNCS, vol.5284, pp.341–356, Springer, Heidelberg, 2008.

Minoru Kuribayashi received the B.E., M.E., and D.E. degrees from Kobe University, Kobe, Japan, in 1999, 2001, and 2004 respectively. From 2002 to 2007, he was a Research Associate in the Department of Electrical and Electronics Engineering, Kobe University. Since 2007, he has been an Assistant Professor at Division of Electrical and Electronics Engineering, Kobe University. His research interests are in digital watermark, information security, cryptography, and coding theory. He is a member of IEEE.

Masakatu Morii received the B.E. degree in electrical engineering and the M.E. degree in electronics engineering from Saga University, Saga, Japan, and the D.E. degree in communication engineering from Osaka University, Osaka, Japan, in 1983, 1985, and 1989, respectively. From 1989 to 1990 he was an Instructor in the Department of Electronics and Information Science, Kyoto Institute of Technology. From 1990 to 1995 he was an Associate Professor at the Department of Computer Science, Faculty of Engineering at Ehime University. From 1995 to 2005, he was a Professor at Department of Intelligent Systems and Information Science, Faculty of Engineering, University of Tokushima. Since 2005, he has been a Professor at the Department of Electrical and Electronics Engineering, Kobe University. His research interests are in error correcting codes, cryptography, discrete mathematics and computer networks. He is a member of the IEEE, the Japan Society for Industrial and Applied Mathematics, the Information Processing Society of Japan and the Society of Information Theory and Its Applications.