Efficient Anti-Scalable Anti-Collusion Fingerprinting

Vincent Havyarimana, Qiaoliang Li

International Journal of Advancements in Computing Technology, Volume 3, Number 3, April 2011

Abstract

Digital fingerprinting schemes are techniques to protect the copyright of digital contents. One of the important problems is a collusion attack such that several users combine their copies of a same content to modify or delete the embedded fingerprint. Trappe et al. proposed the AND anti-collusion codes (AND-ACC) against the average attack. However, the scheme is vulnerable to linear combination attack (LCCA) and cannot support a large number of users. For this last issue, Seol and Kim highlighted a code (called SK) by extending the AND-ACC. Unfortunately, that code is weak against majority attack and LCCA. In this paper, we improve the SK scheme by adding to each group the subcode.

The new code can resist average attack when the inter-group collusion is less likely than intra-group collusion due to geographic conditions. Non-blind detection statistics with the knowledge of the host and soft-threshold detection are used to identify colluders within each guilty group. Our model increases the probability of tracing \( O(n \log^{-1}(N)) \) colluders within each guilty group, where \( N \) is the fingerprinting length and \( n \) is the number of users in each group. Experiment results on the real images show that our code is robust to average attack.

Keywords: Group Subcode (GSC), User Subcode (USC), Average Attack, Soft-thresholding Detection, Guilty Group, Colluders

1. Introduction

As the multimedia technology progresses, the digital representation of multimedia contents becomes popular. Digital fingerprinting is the technique of embedding a set of marks into a host signal to produce a set of fingerprinted signals that each appears identical for use, but different from one another in their representation. However, colluders may gather their copies and make a new copy to avoid being identified; this is known as collusion attack. To design a specific code that resists the collusion attack, Boneh and Shaw [1] proposed a collusion-secure (CS) code based on the principle of marking assumption. To improve the performance of the CS code, Trappe et al. [2] introduced the AND anti-collusion codes (AND-ACC) based on orthogonal code modulation by using bit complement of the incident matrices of balanced incomplete block designs (BIBD). To support large number of users, Seol and Kim presented a code (called SK) by extending the AND-ACC and added a secret random sequence \( \lambda \) with a repetition constant \( M \) [3]. However, Wu and Zhao focused on a joint LCCA-Majority attack and extended LCCA to attack SK scheme [4].

In this paper, we enhance the SK scheme by adding to each group a group subcode (GSC) and apply the collusion attack. The collusion attack is modeled as averaging different copies followed by an additive noise.

Our main objective is to identify the guilty groups and then trace \( O(n \log^{-1}(N)) \) colluders within each guilty group, where \( N \) is the fingerprinting length and \( n \) is the number of users in each group. The scheme is built under the assumption that the inter-group collusion is less likely than intra-group collusion due to geographic conditions.

The rest of this paper is organized as follows: In Section 2, the proposed scalable anti-collusion fingerprinting is highlighted. Collusion attack and tracing strategy are introduced in Section 3. Section
Efficient Anti-Scalable Anti-Collusion Fingerprinting
Vincent Havyarimana, Qiaoliang Li
International Journal of Advancements in Computing Technology, Volume 3, Number 3, April 2011

4 shows the analysis of experimental results. Conclusion and some suggestions for further work are finally drawn in Section 5.

2. The proposed scalable anti-collusion fingerprinting

To construct the scheme, we divide the code into groups. The so-called "user subcode" (USC) is assigned to each user of a group. We then add data which represent the Group Information (GI) to each group. This GI is in turn called "group subcode" (GSC). Each user within each group is characterized not only by his/her USC but also by his/her own information from a group subcode. The SK (Seol and Kim) code as proposed by Wu and Zhao in [4] inspires us to define the user subcode. Indeed, each USC contains $K$ symbols. All users are divided into $L$ groups and each group represents $n$ watermarks $\{w_{1k}, w_{2k}, \ldots, w_{nk}\}$ that are fingerprints associated with different users.

Thus, if $K = 16$, the scheme is arranged such as each USC of user $U_j^l$ is given by

$$W_j^l = \{w_{j1}, w_{j2}, \ldots, w_{jk}, \ldots, w_{jk}\}$$

where $w_{jk} \in G^l$ if $k = \left\lfloor \frac{J}{n} \right\rfloor$ and otherwise $w_{jk} = \lambda$. In this case, $l = 1, 2, \ldots, L$ is the number of groups and $j = 1, 2, \ldots, n$ is the number of users in each group. Therefore, our scenario contains $M = L \times n$ users.

Although $\lambda$ is similar to an anti-collusion code, it has not been taken into consideration in our model. Indeed, it plays a great role especially in minority or majority attack when for example the bits are embedded with $\lambda$ by minority or majority decision.

To define the GSCs of our model, it is required that each group has its own GSC different from others. We totally need $L$ GSCs for $L$ groups.

**Embedding technique**: In our scheme, we embed both the GSC and USC into the host signal by mapping them to spreading sequences. Hence, the fingerprint of user $j$ who belongs to group $l$ is defined as follows

$$e_j^l = (1 - \rho)w_j^l + \rho b_j^l$$

where $\rho \in [\frac{1}{2}, 1]$ is used to adjust the relative energy between the GSC and USC. A higher $\rho$ puts more energy on GI and thus provides a more accurate detection of the guilty groups [5].

Moreover, $w_j^l$ and $b_j^l$ represent "USC" and "GSC" of user $j$ respectively; $w_j^l$ is a watermark of user $j$ who belongs to group $l$ such as $w_j^l = \sum_{m=1}^{n} c_{mj} u_m$ with $c_{mj} \in \{0, 1\}$; all $u_m (m = 1, \ldots, n)$ constitute an orthonormal basis. The matrix $C = (c_{mj})$ (Appendix A) is a derived code matrix, where each column of $C$ contains a derived codevector for a different user [2]. In turn, $b_j^l = \sum_{m=1}^{n} r_{mj} u_m$ with $r_{mj} \in \{0, 1\}$, where $R = \{r_{mj}\}$ is an expanded Redheffer matrix. Indeed, in 1977, R. Redheffer described a matrix that is closely connected to the Riemann Hypothesis (RH). It is the $n \times n$ matrix whose $(i, j)$ entry is 1 if $i | j$ (i divides j) or if $j = 1$, and otherwise is 0, for $1 \leq i, j \leq n$ [6].

Equation (2) becomes

$$e_j^l = \sum_{m=1}^{n} d_{mj} u_m$$
where $d^l_{mj} = (1 - \rho)\epsilon^l_{mj} + \rho R^l_{mj}$. A user $j$ within a group $l$ is identified by $e_j'$ or $d_j'$. We assign the bits $d^l_{mj}$ to different fingerprint in a matrix $D^l = \{d^l_{mj}\}$ called a derived codevector, where each column of $D^l$ contains a derived codevector for a different user $j$. Note that $d^l_{mj} = 1$, if the $m^{th}$ element of $d_j'$ is greater than zero and otherwise $d^l_{mj} = 0$. Example of $D^l = \{d^l_{mj}\}$ is shown in Appendix A. Each group represents a single AND-ACC code and produces a fingerprinting set as $\{e_1', e_2', \ldots, e_{20}'\}$.

Suppose that our model contains 16 groups ($L = 16$), to build GSCs of different groups, the matrix $R_{16:20}$ given below is arranged such as for groups of odd order, elements of all corresponding row-vectors are all 1’s whereas for groups of even order, elements of all corresponding row-vectors are all 0’s; thus,

For $l = 1$, elements of the first row-vector of $R_{16:20}$ are all 1’s;
For $l = 2$, elements of the second row-vector of $R_{16:20}$ are all 0’s;
For $l = 3$, elements of the third row-vector of $R_{16:20}$ are all 1’s;
For $l = 4$, elements of the fourth row-vector of $R_{16:20}$ are all 0’s;

For $l = 15$, elements of the fifteenth row-vector of $R_{16:20}$ are all 1’s;
For $l = 16$, elements of the sixteenth row-vector of $R_{16:20}$ are all 0’s;

\[
\begin{bmatrix}
11111 & 11111 & 11111 & 11111 & 10000 \\
11010 & 10101 & 01010 & 11111 \\
10100 & 10010 & 01001 & 11111 \\
10010 & 00000 & 01000 & 10000 \\
10001 & 00000 & 00001 & 11111 \\
10000 & 10000 & 01000 & 10111 \\
10000 & 01000 & 00100 & 10000 \\
10000 & 00100 & 00010 & 11111 \\
10000 & 00010 & 00001 & 11111 \\
10000 & 00000 & 00000 & 00000
\end{bmatrix}
\]

\[
\begin{bmatrix}
11111 & 11111 & 11111 & 11111 & 10000 \\
11010 & 10101 & 01010 & 11111 \\
10100 & 10010 & 01001 & 11111 \\
10010 & 00000 & 01000 & 10000 \\
10001 & 00000 & 00001 & 11111 \\
10000 & 10000 & 01000 & 10111 \\
10000 & 01000 & 00100 & 10000 \\
10000 & 00100 & 00010 & 11111 \\
10000 & 00010 & 00001 & 11111 \\
10000 & 00000 & 00000 & 00000
\end{bmatrix}
\]

\[
\begin{bmatrix}
11111 & 11111 & 11111 & 11111 & 10000 \\
11010 & 10101 & 01010 & 11111 \\
10100 & 10010 & 01001 & 11111 \\
10010 & 00000 & 01000 & 10000 \\
10001 & 00000 & 00001 & 11111 \\
10000 & 10000 & 01000 & 10111 \\
10000 & 01000 & 00100 & 10000 \\
10000 & 00100 & 00010 & 11111 \\
10000 & 00010 & 00001 & 11111 \\
10000 & 00000 & 00000 & 00000
\end{bmatrix}
\]

**Example of some matrices:** (a) Expansion of Redheffer matrix $R_{16:16}$ into the matrix $R_{16:20}$ using Matlab 7.0.1.; (b) GI 1 whose elements of the first row-vector are all 1’s; (c) GI 16 whose elements of the sixteenth row-vector are all 0’s

According to the additive embedding method [7], the fingerprinted copy of each user $j$ who belongs to group $l$ is given by

\[
y_j' = x_j' + \alpha e_j'
\]

where $\alpha$ is a scaling factor that controls the energy of the embedded watermark and uses the just-noticeable-difference (JND) from human visual system models [8].
3. Collusion attack and tracing strategy

In our model, we assume that inter-group collusion is less likely than intra-group collusion due to geographic conditions: users from Canada can easily make collusion attack with their country-mates but difficultly even impossible with users from Burundi (in Africa). However, since few colluders in each group would be willing to take higher risk than others, they generally would make contributions of an approximately equal amount in the collusion. In this paper, we deal with intra-group collusion. Due to its effectiveness [9], we highlight the collusion attack based on $T$-user averaging collusion plus additive noise. A number of other collusions based on order statistics, such as minimum collusion attack, have been shown to be well approximated by such a model [10].

From the equation (2), the suspicious copy $\hat{Y}$ becomes:

$$\hat{Y} = T^{-1} \sum_{l=1}^{L} \sum_{j \in S'} y_j^l + d = aT^{-1} \sum_{l=1}^{L} \sum_{j \in S'} e_j^l + x + d$$

(3)

Here, $x = x(I)$ and $L$ are respectively the host signal and the total number of groups; $S' \subseteq \{1, \cdots, n\}$ is a subset with size $|S'| = t'$ and contains the members of group $l$ who contribute in the collusion attack, $t'$'s satisfy $\sum_{l=1}^{L} t' = T$; $n$ is the number of users in each group. The additional distortion is modeled as an i.i.d. (independent and identically distributed) additive Gaussian noise $d = d(l)$ with zero-mean and variance $\sigma_d^2$.

In our tracing strategy, the detection scheme consists of two levels:

3.1. Identification of groups in which colluders are from

Our objective is to determine for each $l (l = 1, 2, \ldots, L)$ whether group $l$ is one of the guilty groups ($\hat{L}$). This can be formulated as a binary hypothesis testing problem, with $H_0$ and $H_1$ corresponding to the hypotheses of $l \not\in \hat{L}$ and $l \in \hat{L}$, respectively. The presence of 1’s and 0’s in $D'$ plays a great role in our scheme because not only each user within group $l$ is identified by $e_j$ or $d_j$ but also we are under the anti-collusion fingerprinting scheme in which the logical AND operation contributes more. Referring on [2], a class of binary-valued anti-collusion codes has been proposed, where the shared bits within code vectors allow for the identification of colluders. Indeed, a group with an important number of 1’s is supposed to have a high probability of containing more colluders and therefore easy to be detected.

The probability of group $l$ to participate in average collusion attack is determined by

$$P_G(l, \cdots, L) = \left(\frac{\left\|\hat{y} \times \text{weight}(R')\right\|}{\left(\sum_{l=1}^{L} N \cdot \sigma_d^2 \right)^{1/2} \cdot \text{weight}(R')}\right)^{-1}$$

(4)

where $\text{weight}(R')$ is a number of 1’s in matrix $R'$, which in turn represents GSC of group $l$; $R'$ is matrix whose elements are all 1’s with the same size as $R'$; $\left\|\hat{y}\right\|$ is the Euclidean norm of a vector $\hat{y}$ where $\hat{y}$ is a colluded copy gotten by extracting the GSCs from any group and $N$ is the fingerprinting length. A group $l \in \hat{L}$ if $P_G(l) \geq h$, where $h$ is a predetermined threshold. In order to provide a good indicator of a confident result [11], $h = 0.75$ and therefore, the experiment results in Figure 1 show that only groups $l = 7, 9, 11, 13, 15$ are successfully considered guilty.
Efficient Anti-Scalable Anti-Collusion Fingerprinting
Vincent Havyarimana, Qiaoliang Li
International Journal of Advancements in Computing Technology, Volume 3, Number 3, April 2011

Figure 1. Identification of guilty groups with \( h = 0.75 \): \( \hat{L} = \{7, 9, 11, 13, 15\} \)

3.2. Identification of colluders within each guilty group

After detecting the guilty groups, we identify colluders within each guilty group separately given that inter-group is less likely than intra-group collusion. In each guilty group \( l \), the suspicious copy \( Y \) becomes:

\[
Y' = (f')^\top \sum_{j \in S} \epsilon'_j + x' + d
\]

For better detection performance [12], we consider a non-blind detection scenario where the host signal is first removed from the test copy before colluder identification. It is then preferable to use the normalized correlator vector \( T'' \) because of the orthogonality of \( [u_m] \). Hence, the detection statistic within each group \( l \), with \( m^{th} \) component is given by

\[
T'''(m) = \left( Y' - x' \right]^\top \cdot u_m \left( \sigma_d^2 \cdot \|u_m\|^2 \right)^{-1/2}, m = 1, \ldots, v
\]

where \( x' \) is the host signal of the guilty group \( l \), \( \|u_m\| \) the Euclidean norm of \( u_m \)’s such as \( \|u_m\| = \|u\| \) and \( \sigma_d^2 \) is the variance of additive Gaussian noise. In this case,

\[
T''' = \beta (f')^\top D' \Phi' + n
\]

Here, \( \beta = \alpha \left( \|u\|^2 (\sigma_d^2)^{-1} \right)^{1/2} \) is assumed known, \( t' \) is the number of 1’s in \( \Phi' \) and \( n = \left( u_1, \ldots, u_l, \cdot \right) \| u \|^2 \sigma_d^2 \left( \|u\|^2 \right)^{-1/2} \) follows \( N(0, I_v) \) distribution. Equation (6) can also be written as

\[
f(T''' \mid \Phi') = N \left( \beta (f')^\top D' \Phi', I_v \right)
\]

In order to estimate \( \Phi' \) in each guilty group \( l \), we use an Adaptive Sorting Algorithm (soft-
thresholding detection) proposed in [2]. Indeed, after computing $T_{N}^{l}$, this algorithm sorts at first elements of $T_{N}^{l}$ in descending order and records corresponding index vector as $I$.

By initializing $\Phi^{i}=1$ (1 is the $n$-dimensional vector consisting of all ones) and $m=0$, the likelihood is determined according to the formula (8). Given that $i$ is the order of $I(m)$ and $\Gamma(I)=1$, we define $d_{i}$ as the $i^{th}$ row of $D^{l}$ (the derived code matrix of guilty group) and use the fact that the element-wise multiplication $\cdot$ of the binary vectors corresponds to the logical AND operation. If the likelihood of largest statistic $T_{N}^{l}(m)$ is greater than the next one (in descending order), $\Phi^{i}$ is computed. $\Phi^{i}$ is updated iteratively via likelihood of $T_{N}^{l}$. Hence, each column vector of $\Phi^{i} \in \{0,1\}^{n}$ indicates colluders via the location of components of $\Gamma$ whose values are 1’s.

4. Analysis of experimental results

4.1. Colluder detection within each guilty group

In our experiment, the simulation environment is Matlab7.0.1 software. We choose $10^{4}$ as a fingerprinting length and examine the detection performance at watermark-to-noise-ratio (WNR) of 0 dB. We also use the just-noticeable-difference (JND) from human visual models [8] to control the energy and achieve the imperceptibility of the embedded fingerprint. The distortion $d$ is modeled as an i.i.d Gaussian $\mathcal{N}(0,1)$. There are totally 16 groups ($L=16$) and 20 users ($n=20$) in each group. In addition, $\rho=0.7$ and $\alpha=1$ which are the relative energy between the GSC and USC and scaling factor that controls the energy of the embedded watermark respectively.

Figure 1 shows that only five ($\hat{L}=[7,9,11,13,15]$) out of sixteen groups are successfully considered guilty. We then focus our attention on these identified guilty groups and track $l^{i}=5$ colluders who participate in average attack within each guilty group ($l \in \hat{L}$).

To track those colluders, a non-blind detection statistics with the knowledge of the host is used. Referring on an Adaptive Sorting Algorithm [2] which detects colluders via the location of components of $\Gamma$ whose values are 1’s, the simulation results prove that in groups

\[
\begin{align*}
I=7, \quad & \Phi^{7} = [11000 \ 00000 \ 01000 \ 0001]\; ; \\
I=9, \quad & \Phi^{9} = [0000 \ 10000 \ 00100 \ 10100]\; ; \\
I=11, \quad & \Phi^{11} = [00000 \ 10000 \ 00001 \ 01010]\; ; \\
I=13, \quad & \Phi^{13} = [11000 \ 00100 \ 00001 \ 00001]\; ; \\
I=15, \quad & \Phi^{15} = [10100 \ 00010 \ 00000 \ 00001] .
\end{align*}
\]

Therefore, the sets of users who successfully make average collusion are respectively

\[
\begin{align*}
\{U_{1}^{7}, U_{2}^{7}, U_{12}^{7}, U_{19}^{7}, U_{20}^{7}\} , \\
\{U_{1}^{9}, U_{9}^{9}, U_{13}^{9}, U_{16}^{9}, U_{18}^{9}\} , \\
\{U_{1}^{11}, U_{5}^{11}, U_{12}^{11}, U_{17}^{11}, U_{19}^{11}\} , \\
\{U_{1}^{13}, U_{2}^{13}, U_{6}^{13}, U_{15}^{13}, U_{20}^{13}\} , \\
\{U_{1}^{15}, U_{3}^{15}, U_{8}^{15}, U_{12}^{15}, U_{19}^{15}\} .
\end{align*}
\]

4.2. Experiment results on real images

In order to demonstrate the performance of our scheme, we divide the original host image into
8 x 8 blocks and the fingerprint is perceptually weighted and then embedded into the block DCT coefficients [13, 14]. To generally represent the performance, the 512 x 512 Lena is chosen as the host image. Figure 2 describes images obtained after average collusion attack in different guilty groups via their respective peak signal-to-noise ratio (PSNR).

![Figure 2. Original image (a); images (Lena) (b), (c), (d), (e), and (f) obtained after average collusion attack within group 7, group 9, group 11, group 13, and group 15 respectively](image)

Theoretically, PSNR is used to analyze quality of image, sound and video files and expressed in dB (decibels). PSNR calculation of two images, one original and an altered image, describes how far two images are equal. The following is the PSNR formula:

\[
PSNR(dB) = 10 \times \log\left( \frac{255^2}{MSE} \right),
\]

\[
MSE = \sum_{i} \sum_{j} (A_{ij} - B_{ij})^2 (p \times q)^{-1},
\]

In equation (9), MSE represents Mean-Square error; \( x \) and \( y \) are respectively the width and height of the image; \( p \times q \) is the number of pixels (or quantities); \( A \) and \( B \) indicate the original and altered (or averaged) images.

We note that if PSNR is high, the two images are similar. However, if PSNR is low, the original and altered images are quite different. The larger is the PSNR; the better is the quality of the image.

Generally, if PSNR value is greater than 35dB the watermarked (or fingerprinted) image is within acceptable degradation levels, i.e. the watermarked is almost invisible to human visual system [15]. Moreover, the experiment results indicate that there is no perceptive damage to original image. This means that the original image is similar to the image after average collusion attack.

Finally, due to the similarity between the original and the averaged image for Lena images, we conclude that the proposed scalable anti-collusion fingerprinting is robust to the average collusion attack.

5. Conclusion and suggestions for further work

In this paper, we investigated the problem of the scalable anti-collusion fingerprinting that resists the average collusion attack followed by an additive Gaussian noise. We proposed the tracing strategy not only for identifying guilty groups but also colluders within these suspicious groups. Our proposed
scheme was defined such that inter-group collusion was less likely than intra-group collusion due to geographic conditions. To each group of users, we added a group subcode (GSC). For this purpose, we introduced and re-arranged an expanded Redheffer matrix. We then detected guilty groups by using probability method and the experiment results indicated that five out of sixteen groups were effectively considered guilty.

In order to identify colluders within each guilty group, the non-blind detection was taken into consideration. By applying a soft-thresholding detection based on an Adaptive Sorting Algorithm, the experiment results showed that five out of twenty colluders could be successfully located. The performance of our scheme was proved when using the real images (Lena): For each averaged image, the value of the peak signal-to-noise ratio (PSNR) was greater than 35 dB. The quality of the images was better and therefore no perceptive damage to original image. This demonstrates that our code was robust to average collusion attack.

Presently, our paper was limited on the average attack. The linear combination collusion attack (LCCA) could be considered in the further investigation. Moreover, suppose that users were in the same region where inter-group collusion could occur. It would be better for interested researchers to exploit the behavior of the proposed scheme referring on this last situation by applying average attack or LCCA.

6. Acknowledgement

This research work is partially supported by the NSF of China (Grant No. 11071272 and 10571052)

7. References

Appendix A

Based on the derived code matrix $C = (c_{mj})$ (a) proposed in [2] and if $\rho = 0.7$, we can constitute $D^1 = (d_{mj})$ as follows:

$$d_{mj}^1 = 0.3c_{mj} + 0.7r_{mj}^i = 0.3\times\begin{bmatrix}0000111111111111\end{bmatrix}^T + 0.7\times\begin{bmatrix}11111111111111\end{bmatrix}^T = \begin{bmatrix}0.70.70.71\end{bmatrix}^T = \begin{bmatrix}1111111111\end{bmatrix}^T$$

$$d_{mj}^2 = 0.3c_{mj} + 0.7r_{mj}^i = 0.3\times\begin{bmatrix}0111000111111111\end{bmatrix}^T + 0.7\times\begin{bmatrix}1000000000000000000\end{bmatrix}^T = \begin{bmatrix}0.710.30.0000.30.30.30.30.30.3\end{bmatrix}^T = \begin{bmatrix}1111001111111\end{bmatrix}^T$$

$$d_{mj}^{16} = 0.3c_{mj} + 0.7r_{mj}^i = 0.3\times\begin{bmatrix}111111111100001011\end{bmatrix}^T + 0.7\times\begin{bmatrix}1111111111111111111\end{bmatrix}^T$$

$^T$ is the transposition of function of a given matrix or simply a vector matrix.

The procedure to determine the matrices $D^2, \ldots, D^{16}$ is the same.

\[
\begin{bmatrix}
0000 & 1111 & 1111 & 1111 & 1111 \\
0111 & 1111 & 0001 & 1111 & 1111 \\
0111 & 1111 & 1110 & 0011 & 1111 \\
0111 & 1111 & 1101 & 1101 & 1111 \\
0111 & 1111 & 1101 & 0111 & 1111 \\
0111 & 0111 & 1110 & 1110 & 1110 \\
0111 & 1110 & 0111 & 1110 & 1110 \\
1101 & 1110 & 0111 & 1110 & 1110 \\
1101 & 1101 & 1110 & 1110 & 1110 \\
1101 & 1101 & 0111 & 1110 & 1110 \\
1101 & 1101 & 1110 & 0111 & 1110 \\
1101 & 1101 & 1110 & 1110 & 1110 \\
1101 & 1101 & 1110 & 1110 & 1110 \\
1110 & 1110 & 1110 & 1110 & 1110 \\
1110 & 1110 & 1110 & 1110 & 1110 \\
1110 & 1110 & 1110 & 1110 & 1110 \\
1110 & 1110 & 1110 & 1110 & 1110 \\
1110 & 1110 & 1110 & 1110 & 1110 \\
1110 & 1110 & 1110 & 1110 & 1110 \\
\end{bmatrix}
\]