A Novel Public Watermarking System based on Advanced Encryption System  
Ko-Ming Chan and Long-Wen Chang  
Department of Computer Science  
National Tsing Hua University  
Hsinchu, Taiwan, 300  
lchang@cs.nthu.edu.tw

Abstract  
A lot of digital watermarking techniques have been proposed to resolve the issues of copyright protection. However, almost proposed watermarking methods keep the watermarking algorithm private to ensure the embedded watermark secret. If the watermarking technique needs to be widespread applied to realistic multimedia environment, the algorithm used by watermarking techniques should be public. In this paper, a novel watermarking scheme which can be public is presented. The proposed watermarking technique is developed based on the following criterions:  
(1) the watermarking algorithm is open;  
(2) the embedded watermark can be extracted and embedded by the people who own the secret key. The watermarking scheme employs the advance encryption standard (AES) and the Reed-Solomon code, to make the watermarking algorithm public. Simulation shows that the proposed algorithm can be very robust to resolve the ownership of the digital image.

1. Introduction  
Digital watermarking technique can provides copyright protection for digital data [1-3]. Almost all watermarking methods, which have been proposed today, can provide robust and secret watermark and against various attacks such as A/D or D/A converting, data compression, etc. The watermark is robust and secret due to the owner keeps the algorithm private. If the owner reveals the watermarking algorithm, this is equivalent to tell the location for watermarking, and the malicious attack can easily eradicate or destroy the embedded watermark. Generally speaking, a serviceable watermarking technique should reveal the algorithm to allow all people to manipulate and still provide a robust and secret watermark.

Cryptography is well known to make message secrecy[6-7] and can be applied to the watermarking system to help to reveal the watermarking algorithm without disclosing the location for embedding the watermark. The Rijndael block cipher is the next generation of advanced encryption standard for 21 century [8]. We use it to encrypt the positions for watermarking. Error-Correcting code is capable of providing reconstruction when some of the transmitted codes are lost. Reed-Solomon code [9-11] is adopted for the proposed watermarking method to generate fragile watermark and to correct the damaged image.

2. The proposed watermarking Algorithm  
2.1 The watermark embedding Algorithm  
The proposed watermarking system embeds two different watermarks into the spatial and the frequency domain, separately. In the frequency domain, a robust watermark is embedded first. Then, a fragile watermark is embedded into the LSB plane in the spatial domain. Figure 1 illustrates the watermark embedding procedure. The robust watermark embedding algorithm is stated below:

1. \( W_{RS} = RS\_Encode(W_o) \),
2. \( I_z = Set\_LSB\_Zero(I_o) \),
3. \( CS_0 = DWT(I_z) \),
4. \( MSCPS = Order\_of\_Significant\_Coefficients(CS_0, k, \|W_{RS}\|) \),  
5. \( CS_W = Embed\_Robust\_Watermark(CS_0, WMPS, W_{RS}) \),
6. \( I_{wf} = IDWT(CS_W) \).

In step 1, the original watermark \( W_o \) is first encoded with a Reed-Solomon encoder to generate \( W_{RS} \), where “o” indicates original, “RS” means the Reed-Solomon code is encoded. In step 2, the LSB bit of every pixel of the original image \( I_o \) is set to zero to obtain a new image \( I_z \), where “z” means that the least significant bit is zero. In step 3, \( I_z \) is processed by Discrete Wavelet Transformed to obtain four wavelet sub-bands coefficients LL, HL, LH and HH. \( CS_0 \) is the wavelet coefficient sequence, which records the LL, HL, LH, HH coefficients in raster scan order, respectively. In step 4, major significant coefficients (MCS) are first selected according to the length of encoded watermark, \( \|W_{RS}\| \).
and selection factor $k$, i.e. $k|W_{RS}$| most significant coefficients of the LL band of $CS_0$. Usually $k > 1$, this can ensure the locations, which are selected to embed the robust watermark, to be secret. The order of these major significant coefficients is recorded in the descending order of magnitude to be MSCPS. Each byte records one position. After MSCPS is generated, MSCPS is encrypted with a key $K$ by the Rijndael block cipher to obtain Cipher-MSCPS (C-MSCPS). That is, $W_{RS}$ significant bytes of C-MSCPS are selected as the embedding positions. The position, which corresponds to the most significant byte of C-MSCPS, embeds the first bit of encoded watermark, $W_{RS}$ . The above 6 steps embed robust watermark into the original image in the frequency domain. The watermark is added to the significant coefficients of the LL band of the original image to ensure its robustness.

Figure 2 shows how to generate C-MSCPS and determine the embedded wavelet coefficient positions. After the C-MSCPS is produced, WMPs is generated according to C-MSCPS and $W_{RS}$ . The embedding watermark positions are recorded in the watermark position sequence WMPs. Figure 3 shows how to generate C-MSCPS. MSCPS is encrypted with a secret key $K$ by the Rijndael block cipher to produce C-MSCPS. The fourth block of C-MSCPS with value 66 is the most significant block. Then, the position 3 with wavelet coefficient value 220 of LL band, is chosen to embed the watermark bit. Block 7 of C-MSCPS is the second significant block, and position 73 of LL is chosen to embed the second watermark bit. According to C-MSCPS, the position 7, 8, 9 of LL are chosen to embed third and fourth watermark bits, respectively. In step 5, modify the wavelet coefficients of $CS_0$ according to WMPs and encoded watermark $W_{RS}$ to obtain new wavelet coefficients sequence $CS_W$ . The watermark embedding is illustrated as follows [4, 5]:

1. If $W_{RS}(i) = 0$, then $v_i = v_i(1 + \alpha \ast (-1))$
2. If $W_{RS}(i) = 1$, then $v_i = v_i(1 + \alpha \ast (1))$,

where $W_{RS}(i)$ is the $i^{th}$ watermark bit, $v_i$ is wavelet coefficient value of $CS_0$, and $\alpha$ is a scalar factor and it determines the embedded watermark strength. In step 6, inverse discrete wavelet transformed with the modified wavelet coefficients $CS_w$ to obtain the image $I_w$ , which contains the robust watermark $W_{RS}$ .

The embedding algorithm of fragile watermark is described below:

7. $I_{w^f} = \text{Set_LSB_Zero}(I_{w})$
8. RS-ECC = Generate_RS_ECC ($I_{w^f}$)
9. C-RS-ECC = Rijndael_Encrypt (RS-ECC, K)
10. $I_w = \text{Embed_Fragile_Watermark}(I_{w^f}, \text{C-RS-ECC})$

In step 7, all LSB of $I_{w^f}$ are reset to zero to obtain $I_{w^f}$.

We add the fragile watermark C-RS-ECC in the LSB plane of $I_{w}$ . In step 8, $I_{w^f}$ is encoded by Reed-Solomon encoder to obtain the fragile watermark RS-ECC. The robust watermark C-RS-ECC contains all the parity check bits of encoded Reed-Solomon code and the fragile watermark C-RS-ECC. In step 9, RS-ECC is encrypted with a secret key $K$ and the Rijndael block cipher to obtain the fragile watermark C-RS-ECC. In step 10, all the LSB of $I_{w^f}$ are replaced with C-RS-ECC in a raster scan order and the final watermarked image $I_w$ is obtained.

After above 10 steps, $I_w$ contain two watermarks: a robust watermark $W_{RS}$ and a fragile watermark C-RS-ECC. The robust watermark $W_{RS}$ is used to carry the copyright information and the fragile watermark C-RS-ECC is used to verify the image integrity and capable of providing the ability to recover altered image.

### 2.2 Watermark Extraction Procedure

There are two phases in watermark extraction procedure: the inspecting phase and the extracting phase. In the inspecting phase, the test image $I_w^+$ is inspected whether $I_w^+$ is altered or not. In the extracting phase, the watermark $W_{RS}^+$ is extracted by comparing to the original image $I_0$ . Figure 4 demonstrates the watermark extraction procedure. The extracting algorithm is described below:

#### The Inspecting phase:

1. $(I_{w^f}^+, \text{C-RS-ECC}^+) = \text{Extract_Fragile_Watermark_and_Set_LSB_Zero}(I_w^+)$
2. RS-ECC$^+ = \text{Rijndael_Decrypt}(\text{C-RS-ECC}^+, K)$
3. RS-ECC = Generate_RS_ECC ($I_{w^f}^+$)
4. If RS-ECC$^+ \neq \text{RS-ECC}$, jump to Extracting phase.

#### The Extracting phase:

1. $CS_0^+ = \text{DWT}(I_w^+)$
2. $CS_0 = \text{DWT}(I_z^+)$
3. MSCPS = Order_of_Significant_Coefficients (CS₀, k, \|W_Rs\|).
   C-MSCPS = Rijndael_Encrypt (MSCPS, K),
   WMPS = Select_Watermark_Positions (C-MSCPS, \|W_Rs\|).
4. W₀' = Extract_Robust_Watermark (CS₀', CS₀, WMPS).
5. W₀ = RS_Decode (W₀').

The extracting algorithm is explained below:

**Inspecting phase:**
1. All least significant bits of test image I₀' is collected in a raster scan order to form extracted fragile watermark C-RS-ECC'. After the fragile watermark is extracted, I₀' is obtained by setting all the least significant bits of I₀' to zero.
2. The extracted fragile watermark C-RS-ECC' is decrypted with key K by the Rijndael block cipher to obtain RS-ECC'. The new examining fragile watermark RS-ECC is generated from I₀' with the Reed-Solomon encoder.
3. The embedded fragile watermark RS-ECC' and the new generated fragile watermark RS-ECC are compared. If RS-ECC' is equal to RS-ECC, this means that the received watermarked image I₀' is not altered, i.e. I₀' = I₀. Then, jump to extracting phase. If RS-ECC' is not equal to RS-ECC, i.e. the received watermarked image is altered. Then, jump to step 4.
4. The RS-ECC', which contains the parity check bits of Reed-Solomon code, can help to recover the altered watermarked image.

The **Extracting phase:**
1. Decompose the watermarked image I₀' to obtain the wavelet coefficient sequence CS₀'.
2. Decompose the original I₀ to obtain the wavelet coefficient sequence CS₀.
3. Use the same procedure described in the embedding phase step 4 to obtain the embedded watermark position sequence WMPS.
4. Compare the wavelet coefficients of CS₀' with CS₀ according to watermarking position sequence WMPS.
   (1) W₀'(i) = 1, if v_i' > v_i
   (2) W₀'(i) = 0, if v_i' ≤ v_i
5. Decode W₀' by using the Reed-Solomon decoder to obtain the embedded watermark W₀'.

3. **The simulation Results**
The Tsing Hua University logo of 64 * 64 is used to be the watermark. In order to inspect the extracted watermark, we define a similarity measure to decide whether the watermark exists or not. We define the similarity value between W and W₀' as:
\[
\text{Sim}(W, W₀') = \frac{\sum W_i \oplus W₀'_i}{\sum W_i \oplus W_i},
\]
where \(\oplus\) is the Exclusive-NOR operator, \(1 ≤ i ≤ N \cdot N\), and the similarity value is calculated from total image bits between \(W\) and \(W₀'\). Figure 5 shows the original image and the similarity measures of the extracted watermarks for the watermarked images under various attacks. It shows that the proposed algorithm is very robust.

4. **Conclusion**
A novel watermarking technique that can be public is proposed. The robust watermark is embedded in the wavelet coefficients of LL band of the image to strengthen the watermark and against various attacks such image processing, data compressing, and other malicious modification etc. Advanced encryption standard, the Rijndael block cipher, is adopted to hide the embedded robust watermark positions of the image. The candidate embedded positions are known by the users, who have the secret key to embed/extract the embedded watermark. A fragile watermark is also added in the least significant bit plane of the watermarked image. The fragile watermark, which is encoded with the Reed-Solomon code and the Rijndael block cipher, is capable of providing verification and error correcting abilities.

5. **References**


---

Figure 1: Flowchart for watermark embedding

<table>
<thead>
<tr>
<th>Wavelet Coefficient Value</th>
<th>252</th>
<th>231</th>
<th>224</th>
<th>220</th>
<th>154</th>
<th>140</th>
<th>132</th>
<th>131</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position in LL band</td>
<td>10</td>
<td>33</td>
<td>42</td>
<td>3</td>
<td>65</td>
<td>20</td>
<td>73</td>
<td>41</td>
</tr>
<tr>
<td>MSCPS</td>
<td>7\textsuperscript{th}</td>
<td>5\textsuperscript{th}</td>
<td>3\textsuperscript{rd}</td>
<td>8\textsuperscript{th}</td>
<td>2\textsuperscript{nd}</td>
<td>6\textsuperscript{th}</td>
<td>1\textsuperscript{st}</td>
<td>4\textsuperscript{th}</td>
</tr>
</tbody>
</table>

Figure 2: An example that demonstrates how to generate MSCPS.

<table>
<thead>
<tr>
<th>Rijndael</th>
<th>Key</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSCPS</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>↑</td>
<td>↑</td>
<td>↑</td>
</tr>
<tr>
<td>C-MSCPS</td>
<td>21</td>
<td>45</td>
</tr>
</tbody>
</table>

Figure 3: An example that demonstrates how to generate C-MSCPS.
Figure 4: Flowchart for Watermark extraction

(b) PSNR = 8.527535, Sim = 0.900391
(c) PSNR = 26.734325 Sim = 0.888916
Fig 5 (a) watermarked image without any attacked
(b) Attacked by cropping (c) attacked by sharpening

(d) PSNR = 24.422025 Sim = 0.728271
(e) PSNR = 34.29522 Sim = 0.928223
(f) PSNR = 31.989688 Sim = 0.744385
Fig 5 (d) attacked by uniform noise corruption (e) attacked by blurring (f) attacked by JPEG compression