Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 85 (2016) 31 – 38

Procedia Computer Science

International Conference on Computational Modeling and Security (CMS 2016)

Digital image steganography using variable length group of bits substitution

Gandharba Swain*

Department of Computer Science & Engineering, KL University, Vaddeswaram-522502, India

Abstract

This paper presents two new steganography methods in spatial domain. The basic idea is the substitution of a group of bits in a pixel by another group of bits of same length to hide one or two bits of secret data. The number of bits selected for substitution in a pixel depends upon some pre-defined conditions. The first method hides one bit per pixel and the second method hides two bits per pixel. Although a group of bits are substituted in a pixel, but the maximum change in a pixel value is not more than 2. The security has been improved by making the group length variable for different pixels. In most of the cases the pixel value remains same, but it hides one or two bits of secret data. The experimental results are compared with other methods and found to be satisfactory. The security has been evaluated and found to be improved.

© 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of the Organizing Committee of CMS 2016

Keywords: Steganography, LSB substitution, group of bits substitution, pixel value substitution, data hiding;

1. Introduction

Steganography is a technique for covert communication. It can be done with image, audio, and video carriers [1]. Image steganography techniques can be classified into two major categories such as spatial domain techniques and frequency domain techniques. The image in which secret message is hidden is called as the stego-image. By adding the unnatural message inside a natural image, there is a change in statistics, but if this change is so small, then it can not raise any suspicion [2]. When hiding information inside images usually LSB substitution method is used. But it is vulnerable to simple attacks. Some improvements and modifications have been proposed to strengthen this technique. Mathkour et al. [3] proposed a spiral based LSB substitution approach for hiding message in image. This approach is to divide the image into many segments and apply a different processing on each segment. A technique has been proposed to embed in LSBs of darkest and brightest pixels of an image to improve the security [4]. The LSBs of all the pixels can be supposed as an array and the secret message can be embedded at a maximum matching portion of the array, so that there will be minimum distortion [5,6]. Furthermore, by mapping the words on the LSB array and embedding at maximum matched locations can boost the security to a greater level [7]. To enhance the security and capacity one can hide adaptive number of bits in different pixels by measuring the embedding depth using some statistical analysis [8]. Jain and Ahirwal proposed an adaptive embedding approach by using a private stego-key [9]. The private stego-key consists of five gray level ranges that are selected randomly in the range from 0 to 255. The selected key shows the five ranges and each range substitute different number of bits at LSBs. This technique can embed upto 4 bits in a pixel. Swain and Lenka [10] proposed message bit dependent embedding, wherein the embedding locations in a pixel are selected depending on the bit pattern of the secret message.

Corresponding author Tel: +91-9573975571
E-mail address: gswain1234@gmail.com

1877-0509 © 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of the Organizing Committee of CMS 2016
doi:10.1016/j.procs.2016.05.173

A new track in image steganography called pixel value differencing (PVD) has been proposed by Wu and Tsai [11] in 2003. As per this technique a difference value d is calculated from every non-overlapping block of two consecutive pixels and it is substituted by a new difference value with the bits of secret message being embedded in it. A PVD technique using tri-way pixel value differencing have been proposed in [12]. Like these pixel-block techniques, other techniques based on correlation of a pixel with its neighboring pixels have also been evolved. Chang and Tseng [13] proposed two-sided, three-sided and four-sided side match methods wherein the correlation of a target pixel with its neighboring pixels is exploited to take embedding decision in the target pixel. Improved versions of side match methods have been proposed in [14]. Tseng and Leng proposed the 2-pixel block PVD with a range table based on perfect square [15]. Shen and Huang [18] proposed a steganography technique based on PVD and exploiting modification directions. They achieved higher hiding capacity and improved imperceptibility.

In this paper a new approach hitherto known as group of bits substitution (GBS) has been proposed. There are two schemes. The first one, called 1-bit GBS scheme hides one bit per pixel and the second one, called as 2-bit GBS scheme hides two bits per pixel. Here one pixel means one byte of the image. The embedding is done by replacing a group of bits in a pixel by another group of bits of same length. The rest of the paper is organized as follows. In section 2 & 3 these schemes are narrated. In section 4 the results are discussed and compared with other methods. Finally, the conclusions are given in section 5.

2. The 1-Bit GBS Method

The embedding procedure is as follows. Convert the image to binary and convert the secret message to binary. Convert the length of secret message to binary with 20 bit length. Append the length at the beginning of the binary message. Now simply you call the binary length plus binary message as *message*. One bit of message should be embedded in one byte of the cover image. Suppose the image length is N bytes and the *message* length is n bits. Then the message can be embedded if $n \leq N$. The embedding procedure comprises of the following steps.

Step1- The binary image, A comprises of bytes A_k, for k= 1 to N, where each A_k is 8 bits in length. Each byte is considered to be one pixel. The message, b comprises of bits b_i , for i= 1 to n, where each b_i is a bit 0 or 1.

Step2- If n>N, then display “The image is smaller and can not be embedded”, otherwise initialize k=1 and i=1 and go to step3.

Step3-Suppose the eight bits of A_k are denoted as $d_1d_2d_3d_4d_5d_6d_7d_8$ and assume that, $D_1=d_1d_2d_3d_4d_5d_6d_7d_8$, $D_2=d_2d_3d_4d_5d_6d_7d_8$, $D_3=d_3d_4d_5d_6d_7d_8$, $D_4=d_4d_5d_6d_7d_8$, $D_5= d_5d_6d_7d_8$, $D_6= d_6d_7d_8$, and $D_7= d_7d_8$.

If the byte D_1 is 11111111 or 00000000; then apply simply one bit LSB substitution. After embedding we get D_1 = 11111111 or 11111110 or 00000000 or 00000001.

Else if D_1 is 01111111 or 10000000;

if b_i is 0, then after embedding set D_1= 01111111, Otherwise, if b_i is 1, then after embedding set D_1=10000000.

Else if D_2 is 0111111 or 1000000;

if b_i is 0, then after embedding set D_2= 0111111, otherwise, if b_i is 1, then after embedding set D_2=1000000.

Else if D_3 is 011111 or 100000;

if b_i is 0, then after embedding set D_3= 011111, otherwise, if b_i is 1, then after embedding set D_3=100000.

Else if D_4 is 01111 or 10000;

if b_i is 0, then after embedding set D_4= 01111, otherwise, if b_i is 1, then after embedding set D_4=10000.

Else if D_5 is 0111 or 1000;

if b_i is 0, then after embedding set D_5= 0111, otherwise, if b_i is 1, then after embedding set D_5=1000.

Else if D_6 is 011 or 100;

if b_i is 0, then after embedding set D_6= 011, otherwise, if b_i is 1, then after embedding set D_6=100.

Elseif D_7 is 01 or 10;

if b_i is 0, then after embedding set D_7= 01, otherwise, if b_i is 1, then after embedding set D_7=10.

Step4- Increment k by 1 and i by 1.

Step5- If $i \leq n$ then go to step3, else go to step6.

Step6- Display “Embedded successfully”.

The extraction procedure is very simple and is the reverse of embedding, described by the following steps. Suppose the binary stego-image is B and the message to be extracted bit by bit from it is m.

Step1- Initialize m to blank and initialize the counter, c=1.

Step2- The binary stego-image B comprises of bytes B_k , for k= 1 to N, where each B_k is one byte. Initialize k=1.

Step3- Suppose the eight bits of B_k are denoted as $d_1d_2d_3d_4d_5d_6d_7d_8$ and assume that, $D_1=d_1d_2d_3d_4d_5d_6d_7d_8$, $D_2=d_2d_3d_4d_5d_6d_7d_8$, $D_3=d_3d_4d_5d_6d_7d_8$, $D_4=d_4d_5d_6d_7d_8$, $D_5= d_5d_6d_7d_8$, $D_6= d_6d_7d_8$, and $D_7= d_7d_8$.

If the D_1 is 11111111 or 00000000 or 11111110 or 00000001, then the extracted bit d_8 is appended to m.

Else if D_1 is 01111111, then the extracted bit 0 is appended m. Else if D_1 is 10000000, then the extracted bit 1 is appended to m.

Else if D_2 is 0111111, then the extracted bit 0 is appended to m. Else if D_2 is 1000000, then the extracted bit 1 is appended to m.

Else if D_3 is 011111, then the extracted bit 0 is appended to m. Else if D_3 is 100000, then the extracted bit 1 is appended to m.

Download English Version:

https://daneshyari.com/en/article/488440

Download Persian Version:

https://daneshyari.com/article/488440

Daneshyari.com