Social Networks Steganography using Unions of Lucas Sequences

N. Aroukatos\textsuperscript{1}, K. Manes\textsuperscript{1}, K. Rigos\textsuperscript{1}, and F. Georgiakodis\textsuperscript{1}
\textsuperscript{1}Department of Informatics, University of Piraeus, Piraeus, Greece

Abstract—Since their introduction, social network sites such as Google+, Facebook, Linkedin and Twitter have attracted millions of users, many of whom have integrated these sites into their daily practices. The above networks assist their users to connect with each other based on shared interests, political views, or activities. Sites also vary in the extent to which they incorporate new information and communication tools, such as mobile connectivity, blogging, and photo or video sharing. The ability of photo sharing gives users the opportunity to use these services to exchange secret information using steganographic methods. In this paper, we examine these possibilities and we propose a unique and safe steganographic method using unions of Lucas sequences.

Keywords: Steganography, LSB, Fibonacci, Lucas, Zeckendorf

1. Introduction

What makes social network sites unique is not that they allow individuals to meet strangers, but rather that they enable users to articulate and make their social networks visible. This can result in connections between individuals that would not otherwise be made. While social networks (SNs) have implemented a wide variety of technical features, their backbone consists of visible profiles that display an articulated list of “friends” who are also users of the system. Image files constitute an important feature of users’ profiles. Most users usually upload a large number of personal images in their profiles. These images can be moments of their daily lives, pictures from holidays e.t.c. Theoretically, it will be easy for a SN user to exchange secret information with a member of the same SN and in many cases with a user outside the SN. The visibility of a profile varies by site and according to the user discretion. Many profiles on Facebook are crawled by search engines, making them visible to anyone. Facebook users who are part of the same “friend network” can view each other’s profile, unless a profile owner has decided to deny permission to those in their “friend network”. Structural variations around visibility and access are one of the primary ways that SNs differentiate themselves from each other. The public display of connections is a crucial component of SNs. The “friends” list contains links to each profile, enabling viewers to traverse the network graph by clicking through the friends lists. On the other hand, Google+ use Google+ Circles. Circles allow users to create and share information with groups of friends the same way as in their real life social circles. In the next chapters, we will present a unique steganographic algorithm using Unions of Lucas sequences, that can be used from SN users to exchange secret data.

2. Previous steganographic methods

During the last decade [3] many scientific groups proposed methods that they use and/or extend the LSB (Least Significant Bit) method [1]. Each byte $x$ from $n \times m$ 8-bit RGB image can be represented by an 8-bit binary word $b_7b_6...b_0$. According to the RGB model, each pixel of the image is encoded by 3 bytes, that is, 3 integers in the interval $[0, 255]$. Each byte $x$ is encoded by an 8-bit binary word $b_7b_6...b_0$, where

$$x = \sum_{i=0}^{7} b_i \cdot 2^i \quad \text{and} \quad b_i \in \{0, 1\}.$$  

The 8th (rightmost) bit in this word holds less significant color information than the rest. This bit and, in many cases, a few bits more (e.g. the 7th or 6th) [4] can be replaced by a desired secret bit and so a new stegoimage is built bit by bit. The difference between the two images (the original and the stegoimage) is virtually indistinguishable by the human eye [2]. Of course, many steganalysis programs can detect and in many cases reveal the secret data. The limitations of capacity and secrecy motivated the researchers to develop and extend the LSB method by introducing new base systems other than the binary system.

In this direction, the Fibonacci method, presented in [10], uses the Fibonacci numbers to encode the pixel values of a given target image. The Fibonacci numbers are given by the linear recurrence relation

$$F_n = F_{n-1} + F_{n-2}, \quad n > 1, \quad \text{with} \quad F_0 = 0 \quad \text{and} \quad F_1 = 1.$$  

According to Zeckendorf’s Theorem, every positive integer $x$ can be uniquely represented as a sum of distinct, nonconsecutive Fibonacci numbers. This sum is called the Zeckendorf representation of $x$ [6]. Equivalently, given that $F_k \leq x < F_{k+1}$, for some $k \geq 2$, we have that

$$x = \sum_{i=1}^{k} w_i F_i + 1,$$

where $w_i \in \{0, 1\}$, $w_k = 1$ and we never have $w_i = w_{i+1} = 1$. The sequence $w_nw_{n-1}\ldots w_1$ is a binary word with no consecutive 1’s and it is called the Fibonacci encoding of $x$. Such binary words are called Fibonacci words.
Obviously, in order to encode all possible pixel values of an 8-bit RGB image, we only need the Fibonacci numbers up to 255, that is the sequence

\[(1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233)\]

consisting of all \(F_n\), where \(2 \leq n \leq 13\). Consequently, each pixel value is encoded by a 12-bit Fibonacci word. In this way, we produce 12 bitplanes for embedding data and so we can increase the stego capacity.

For example, the number 39 has the Zeckendorf representation

\[39 = 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 3 + 1 \cdot 5 + 0 \cdot 8 + 0 \cdot 13 + 0 \cdot 21 + 1 \cdot 34 + 0 \cdot 55 + 0 \cdot 89 + 0 \cdot 144 + 0 \cdot 233\]

and it is encoded by the Fibonacci word 000100010000.

Other scientific groups proposed (using the same main idea) other base systems using prime numbers [7], natural numbers [8] or other more “exotic” sets of numbers [9]. These methods have many benefits while they are easy to implement and work very fast. On the other hand, they have inevitable disadvantages. Once revealed to the public, they cannot provide secrecy to their future users. The reason is that the secrecy of these methods depends on the number set that is revealed.

3. A method using Lucas sequences and a Zeckendorf’s Theorem extension

In this section, we present our proposed method, which improves the previously mentioned methods in two directions: Capacity and security. More specifically, increased steganographic capacity is achieved by introducing virtual bitplanes, via a suitable base system (Figure 1). This base system is not unique as in other methods, it is generated from a large set of possible choices, thus operating as an encryption-decryption key.

Consider the quadratic equation \(x^2 - Px + Q = 0\), where \(P\) and \(Q\) are integers. The discriminant of this equation is \(D = P^2 - 4Q\) and the roots are

\[a = \frac{P + \sqrt{P^2 - 4Q}}{2} \quad \text{and} \quad b = \frac{P - \sqrt{P^2 - 4Q}}{2} .\]

It also applies that \(a + b = P\), \(ab = \frac{1}{4}(P^2 - D) = Q\) and \(a - b = \sqrt{P^2 - 4Q}\).

For \(D \neq 0\), we define

\[U_n(P, Q) = \frac{a^n - b^n}{a - b} \quad \text{and} \quad V_n(P, Q) = a^n + b^n .\]

These sequences are called Lucas sequences [5]. We define the set

\[LU = \{1, 2, 3, 4, 5, 9, 11, 13, 17, 21, 33, 40, 43, 65, 85, 121, 129, 171\} ,\]

that is,

\[LU = (U(1, -2) \cup V(3, 2) \cup U(4, 3)) \cap [1, 255]\]

where

\[U(P, Q) = \{U_n(P, Q) : n = 0, 1, 2, \ldots\} \]

and

\[V(P, Q) = \{V_n(P, Q) : n = 0, 1, 2, \ldots\} .\]

In general, the sets of the \(LU\) are selected using the following rule. We choose randomly from the many members of the Lucas sequences, until Union is “complete”, that is, its members taken in ascending order form a sequence of integers in the closed interval \([1, 255]\) which satisfies the following Theorem.

**Theorem 1**: Let \((a_n)_{n \in \mathbb{N}^*}\) be a strictly increasing sequence of positive integers, with \(a_1 = 1, a_2 = 2\) and \(a_n + a_{n+1} \geq a_{n+2}\) and \(n \in \mathbb{N}^*\). Then, every positive integer \(x\) with \(a_n \leq x < a_{n+1} , n \in \mathbb{N}^*\), can be represented as a
sum of different and nonconsecutive terms of the sequence \((a_n)\), with the restriction that the term \(a_n\) appears in the sum.

This theorem is stated and proved in [9] and it can be considered as an extension of Zeckendorf’s theorem.

According to our Theorem, given a sequence \((a_n)\) satisfying the above requirements, any \(x \in \mathbb{N}\) is represented as

\[
x = \sum_{i=1}^{n} w_i a_i,
\]

where \(w_i \in \{0, 1\}\), \(w_n = 1\) and there is no \(i \leq n - 1\), such that \(w_i = w_{i+1} = 1\). The number \(n\) is the unique positive integer satisfying \(a_n \leq x < a_{n+1}\). Therefore, each representation corresponds to a unique Fibonacci word \(w_n w_{n-1} \cdots w_1\), so that each \(x \in \mathbb{N}\) corresponds to at least one Fibonacci word. By choosing the lexicographically greatest corresponding word, we define an encoding for the elements of \(\mathbb{N}\). This is equivalent to applying recursively the restriction of the Theorem. The implementation for this is trivial and, therefore, the process of encoding and decoding each integer \(x\) is straightforward.

For example, the sequence \((1, 2, 3, 5, 7, 9, 11)\) is a sequence of length 7 which encodes all integers in the interval \([0, 22]\). (Note that 22 is obtained as the maximum sum of nonconsecutive terms of the sequence, i.e., \(22 = 11 + 7 + 3 + 1\).) Following the restrictions of the Theorem, the number 18 is represented as

\[
18 = 11 + 7 \quad \text{or} \quad 18 = 11 + 5 + 2.
\]

These representations correspond to the Fibonacci words

\[
w = 1010000 \quad \text{and} \quad u = 1001010
\]

respectively. Since \(w\) is greater than \(u\), the number 18 is encoded by \(w\).

So, every integer number in \([0, 255]\) can be written as a sum of elements of the set \(LU\).

For example, some representations of the number 55 are:

\[
55 = 1 \cdot 1 + 0 \cdot 2 + 0 \cdot 3 + 0 \cdot 4 + 0 \cdot 5 + 0 \cdot 9 + 1 \cdot 11 + 0 \cdot 13 + 0 \cdot 17 + 0 \cdot 21 + 0 \cdot 33 + 0 \cdot 40 + 1 \cdot 43 + 0 \cdot 65 + 0 \cdot 85 + 0 \cdot 121 + 0 \cdot 129 + 0 \cdot 171
\]

or

\[
55 = 0 \cdot 1 + 1 \cdot 2 + 0 \cdot 3 + 0 \cdot 4 + 0 \cdot 5 + 0 \cdot 9 + 0 \cdot 11 + 1 \cdot 13 + 0 \cdot 17 + 0 \cdot 21 + 0 \cdot 33 + 1 \cdot 40 + 0 \cdot 43 + 0 \cdot 65 + 0 \cdot 85 + 0 \cdot 121 + 0 \cdot 129 + 0 \cdot 171
\]

or

\[
55 = 1 \cdot 1 + 0 \cdot 2 + 0 \cdot 3 + 0 \cdot 4 + 0 \cdot 5 + 0 \cdot 9 + 0 \cdot 11 + 0 \cdot 13 + 0 \cdot 17 + 1 \cdot 21 + 1 \cdot 33 + 0 \cdot 40 + 0 \cdot 43 + 0 \cdot 65 + 0 \cdot 85 + 0 \cdot 121 + 0 \cdot 129 + 0 \cdot 171
\]

The above three sums are encoded respectively by the binary words:

\[
0000010000010000010_1LU, 00000010001000010_{1LU}, \quad 00000001100000001_{1LU}.
\]

As in the case of Fibonacci numbers, we use the

\[
000001000001000001_{1LU}
\]

representation which is the lexicographically greatest. Obviously, the set \(LU\) can be defined as a union of different sets, for different values of \(P, Q\). The user can generate a different set \(LU\) and share it with other users as an encryption-decryption key.

4. Measures and results

We propose this method to use in SN for some reasons. Firstly, by using more bitplanes than previous methods, we can achieve higher stego data capacity. Secondly, by using randomly selected Lucas sequences, we generate each time a different set of \(LU\) numbers and this means more secrecy.

Next, we test our method in various SN and we investigate its effectiveness by applying well known image quality measures. The implementation of our method has been done by our own application (Crypto ver. 1.4). For testing purposes, we use some widely used test images such as lena, baboon, airplane, pepper as well as dozens of pictures taken by smartphones and SLR cameras.

To test the ability of SN to host stegoimages, we have built two user accounts in each service. We choose to test Facebook and Google+ while they are the most widely used. Our testing procedure is as follows: First we upload to a user account a large set of stegoimages of different dimensions (Figure 2) and then we download the images from another user account and make our measurements.

We use pictures of various resolutions (from 1200x1200 to 128x128) and various sizes (3Mbytes to 20 Kbytes). After many days of testing, we have the following conclusions:

In the case of Facebook, the algorithm for image resizing and reconstruction destorts our secret data, making the exchange of secret data worthless. In order to deal with this difficulty, we use small images (about 70-80 kb) and we alter...
In Google+, when the image resolution and the image file size do not exceed a certain threshold, the downloaded stego image is exactly the same with the stego image that we have uploaded. For example, our test shows that a resolution of 512X512 pixel with file size about 100 Kb is safe.

We use two metrics, the Mean Square Error (MSE) and the Peak Signal to Noise Ratio (PSNR) to compare the previous steganographic methods with our method, using images with “qualified in Google+” image resolution and size. The MSE is the cumulative squared error between the stego image and the original image, whereas PSNR is a measure of the peak error. Given two \( m \times n \) monochrome images \( I \) and \( K \), where one of the images is considered a noisy approximation of the other, the MSE is defined as:

\[
MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2
\]

and the PSNR is defined as:

\[
PSNR = 10 \log_{10} \left( \frac{MAX_I^2}{MSE} \right),
\]

where \( MAX_I \) is the maximum possible pixel value of the image \( I \). When the pixels are represented using 8 bits per sample, this is 255. A lower value for MSE means lower image distortion, and as seen from the inverse relation between the MSE and PSNR, this translates to a high value of PSNR.

In the tables 1, 2 and in columns 2, 3, 4 and 5 we see the PSNR measurements for each method. We use 4 bitplanes for data embedding in Test1 image and 7 bitplanes in Test2 image. We can see that our method improves the PSNR value of the stego image, when compared to the previous methods.

In Figure 4, we can see the original image and the generated stego images of the methods LSB, Fibonacci, Prime, LU as they appear from left to right and top-down. We can observe that the LU method results in less image distortion, when higher bitplanes are also used for embedding.

5. Conclusions

As we see, Facebook and Google+ give us the ability to exchange with other users easily and freely our secret data using stegoimages that we have uploaded. Our method using Unions of Lucas sequences is an improvement over the previous steganographic methods which take advantage
of different base systems. It is also very important the fact that our method gives us increased security, because the user can build his own Union of Lucas sequences and therefore generate in some sense a unique steganographic “key”. On the other hand, much research and work must be done in order to apply our method in Facebook more efficiently. We must find a method to increase up to a satisfactory level the ability to retrieve the stego data from the downloaded stegoimage.

References